

---

# Live Forensics on a Windows System: Using Windows Forensic Toolchest (WFT)

---

By Monty McDougal



<http://www.foolmoon.net/security/>  
[monty@foolmoon.net](mailto:monty@foolmoon.net)

Live Forensics on a Windows System -- © 2003-2006 Monty McDougal

1

## **Live Forensics on a Windows System: Using Windows Forensic Toolchest (WFT)**

**By Monty McDougal**

**<http://www.foolmoon.net/security/>  
[monty@foolmoon.net](mailto:monty@foolmoon.net)**

Windows Forensic Toolchest (WFT) and this presentation are  
Copyright © 2003-2006 Monty McDougal. All rights reserved.

# Agenda

- Forensic Fundamentals
  - Traditional Forensics
  - Live Forensics
  - Forensic Response Principles
  - Forensic Response Toolkits
- Windows Forensic Toolchest (WFT)
  - Benefits
  - Usage
  - WFT In Action (Demo)

This Page Intentionally Blank

---

# Forensic Fundamentals

---

This Page Intentionally Blank

# System Forensics

"Gathering and analyzing data in a manner as free from distortion or bias as possible to reconstruct data or what happened in the past on a system [or a network]"

Dan Farmer / Wietse Venema (1999)

Dan Farmer / Wietse Venema are generally credited (1999) with the creation of computer forensics as we know it today. They are also the author of one of the first freeware tools for doing forensics named The Coroner's Toolkit (TCT). While this tool suit has generally been expanded and enhanced by many others, it certainly is the basis of modern computer forensics at least within the \*NIX world.

# Forensic Methodologies

- Traditional Forensics
  - Analyzing a “dead” system that has had its power cord pulled
  - Least chance of modifying data on disk, but “live” data is lost forever
- Live Forensics (Often Incident Response)
  - Methodology which advocates extracting “live” system data before pulling the cord to preserve memory, process, and network information that would be lost with traditional forensic approach
  - Goal is to minimize impacts to the integrity of the system while capturing volatile forensic data

Forensic methodologies, generally fall into two broad camps.

The first is the “pure” pull-the-plug traditional forensic methodology advocated for many years by most of the law enforcement community. This method is great for preserving data on disk, but you lose a lot of volatile data which may be useful. A skillful attacker may never even write their files to disk. A real world example of this is the code red worm.

The second methodology, live forensics, recognizes the value of the volatile data that may be lost by a power down and seeks to collect it from a running system. As any such action will in some minor ways later the system, it is not pure in forensic terms. Many people, including the author of this presentation, feel this is an acceptable tradeoff given the value of the data that can be collected from a running system (with minimal impacts).

# Forensic Volatility

- Forensic Data Collection
  - Collects data to be used for later analysis
  - Live forensics focuses on collecting the most volatile information first
  - Response can then continue further, or you power down for a traditional response
- Order of Volatility
  - Memory
  - Swap File
  - Network Processes
  - System Processes
  - File System Information
  - Raw Disk Blocks

Data on a system has an order of volatility. In general, data from the memory, swap space, network processes, and running systems processes is the most volatile and will be lost on system reboot. Raw file system information and data within the raw disk blocks are generally the least volatile. Whenever you collect data, you want to collect the most volatile first before proceeding to the least volatile. The order of volatility is as follows:

## Order of Volatility

1. Memory
2. Swap File
3. Network Processes
4. System Processes
5. File System Information
6. Raw Disk Blocks

# Traditional Forensics

- Traditional forensics focuses on collecting and analyzing information from "dead" file systems (Read Only)
  - Locating, reviewing, and verifying the integrity of "security relevant" files
  - File system analysis (MAC time, file usage, etc.)
  - OS specific and often time consuming
  - Usually limited to criminal cases

Traditional forensics focuses on learning as much about a "dead" file system as possible. While a full analysis can be time consuming, doing one can reveal a lot about an incident. Often times one of the most revealing things that can be done is a MAC time analysis to reconstruct the events of an attack by the files accessed. While this can certainly be manipulated by a skilled attacker, few go to this depth. In general this type of analysis is limited to criminal cases or for cases where the attacker's means of compromise was unknown and the goal is to determine how they got in.

# Traditional Forensic Tools

- Common tools (All OSES)
  - The Coroner's Toolkit (TCT)
    - <http://www.porcupine.org/forensics/tct.html>
  - The Sleuth Kit (TSK)
    - <http://www.sleuthkit.org/sleuthkit/>
  - Autopsy Forensic Browser
    - <http://www.sleuthkit.org/autopsy/>
  - EnCase (Commercial Tool Suite)
    - <http://www.guidancesoftware.com/>

These are some of the more common “traditional” forensics tools. EnCase is included for completeness – I neither use nor endorse it.



# Live Forensics

- Focuses on extracting and examination of the volatile forensic data that would be lost on power off
- Live forensics is not a “pure” forensic response as it will have minor impacts to the underlying machine’s operating state
  - The key is the impacts are known
- Often used in incident handling to determine if an event has occurred
- May or may not proceed a full traditional forensic analysis

Live forensics is the focus of this talk, but specifically in conjunction with the Windows Forensic Toolchest (WFT). The goal of any live forensics task should be to extract and preserve the volatile data on a system while, to the extent possible, otherwise preserving the state of the system. Additionally, this is often the first step of an incident response scenario where a handler is simply trying to determine if an event has occurred. The benefit of using this approach is you have a forensically sound data collection from which to proceed with a full forensic analysis if the initial analysis indicates one is required.

# Live Forensic Tools

- Common Tools (Win OSes)

<b>arp.exe</b>	<b>hunt.exe</b>	<b>ntlast.exe</b>	<b>reg.exe</b>
<b>attrib.exe</b>	<b>ipconfig.exe</b>	<b>openports.exe</b>	<b>regdmp.exe</b>
<b>auditpol.exe</b>	<b>iplist.exe</b>	<b>pclip.exe</b>	<b>RootkitRevealer.exe</b>
<b>autorunsc.exe</b>	<b>ipxroute.exe</b>	<b>promiscdetect.exe</b>	<b>route.exe</b>
<b>cmd.exe</b>	<b>listdlls.exe</b>	<b>ps.exe</b>	<b>sc.exe</b>
<b>cmdline.exe</b>	<b>mac.exe</b>	<b>psfile.exe</b>	<b>servicelist.exe</b>
<b>dd.exe</b>	<b>mdmchk.exe</b>	<b>psinfo.exe</b>	<b>sniffer.exe</b>
<b>drivers.exe</b>	<b>mem.exe</b>	<b>pslist.exe</b>	<b>streams.exe</b>
<b>dumpel.exe</b>	<b>nbtstat.exe</b>	<b>psloggedon.exe</b>	<b>strings.exe</b>
<b>efsinfo.exe</b>	<b>net.exe</b>	<b>psloglist.exe</b>	<b>tlist.exe</b>
<b>fport.exe</b>	<b>netstat.exe</b>	<b>psservice.exe</b>	<b>uname.exe</b>
<b>handle.exe</b>	<b>netusers.exe</b>	<b>pstat.exe</b>	<b>uptime.exe</b>
<b>hfind.exe</b>	<b>now.exe</b>	<b>psuptime.exe</b>	<b>whoami.exe</b>
<b>hostname.exe</b>	<b>ntfsinfo.exe</b>	<b>pulist.exe</b>	

Above is a list of the most common tools used in a forensic response on a windows system. Most of these are either free downloads or come from Microsoft as part of their OS / Resource kits. The location to acquire these is all documented within the WFT config file. A few have been deprecated and may be difficult to locate, but that is also noted in the config file.

# Forensic Response Principles

- Good forensic / incident response follows some generally accepted principles
  - Maintain forensic integrity
  - Require minimal user interaction
  - Gather all pertinent information to determine if an incident occurred for later analysis
  - Enforce sound data and evidence collection
- A Forensic response “should” be scripted or use a common toolkit to enforce above

Like most things, there is generally a “right way” and a “wrong way” to live forensics. The “right way” will generally exhibit four traits:

- Maintain forensic integrity
- Require minimal user interaction
- Gather all pertinent information to determine if an incident occurred for later analysis
- Enforce sound data and evidence collection

One of the keys to any such response, is that it be consistent and verifiable. Therefore, it is highly recommended that the response be automated. There are a number of common toolkits which will assist with this on a windows system. These are covered on the next slide.

# Forensic Response Toolkits

- Common Tools (Win OSes)
  - Windows Forensic Toolchest (WFT)
    - <http://www.foolmoon.net/security/>
  - Incident Response Collection Report (IRCR)
    - <http://tools.phantombyte.com/>
  - First Responder's Evidence Disk (FRED)
    - [http://www.csa.syr.edu/Jesse\\_Kornblum.pdf](http://www.csa.syr.edu/Jesse_Kornblum.pdf)

These are probably three most common forensic response toolkits for Windows. I am of course biased towards Windows Forensic Toolchest (WFT). IRCR has been completely rewritten (borrowing from WFT) and is considerably more powerful than it was when I wrote my original paper and WFT v1.0. If you are looking for alternative to WFT, this is the one I would recommend. FRED, has a slightly different goal than IRCR or WFT, but it may be useful to some people.

---

# Windows Forensic Toolchest (WFT)

---

This Page Intentionally Blank

# Windows Forensic Toolchest (WFT)

- WFT automates live forensics / incident response
  - Many people use it for auditing as well
- Runs a series of tools to collect forensically useful information from Windows NT/2000/XP/2003 machines
- Concept similar to TCT's Graverobber
  - Or a more powerful IRCR (for Windows)

The Windows Forensic Toolchest (WFT) was written to provide an automated incident response [or even an audit] on a Windows system and collect security-relevant information from the system. It is essentially a forensically enhanced batch processing shell capable of running other security tools and producing HTML based reports in a forensically sound manner. A knowledgeable security person can use it to help look for signs of an incident (when used in conjunction with the appropriate tools). WFT is designed to produce output that is useful to the user, but is also appropriate for use in court proceedings. It provides extensive logging of all its actions along with computing the MD5 checksums along the way to ensure that its output is verifiable. The primary benefit of using WFT to perform incident responses is that it provides a simplified way of scripting such responses using a sound methodology for data collection.

The author of this tool is open for suggestions, criticisms of this tool, or offers to help improve the tool's config file and/or its documentation. Comments relating to WFT can be sent to the author at [wft@foolmoon.net](mailto:wft@foolmoon.net).

WFT and the GCFA practical paper which discuss it are available from:

<http://www.foolmoon.net/security/>

Windows Forensic Toolchest (WFT) and this presentation are  
Copyright © 2003-2006 Monty McDougal. All rights reserved.

# Benefits of WFT

- Provide a response that is:
  - Consistent and verifiable
  - Forensically sound methodology
    - Minimizes system impacts\*
    - Enforces known binaries
    - Extensive logging
    - Checksums everything
  - Visually appealing (HTML reporting)

\* Windows Forensic Toolchest (WFT) treads very, very lightly on the system it is being run on (i.e. uses running memory and reads a couple registry entries because it is compiled with Visual C++, but not much else). The tools WFT is invoking do not always exhibit such constraint. The tools included in the default configuration file do not make any “significant” alterations of the system they are being run on. This is described in more detail in the author’s GCFA practical

=====

WFT was designed with forensic principles in mind. As such it is carefully coded, statically compiled, and written to ensure it provides extensive enough logging to be useful even in a court of law (complete with visually appealing reporting).

WFT v2.0 is a complete from the ground rewrite of WFT v1.0. In addition to several code optimizations, version 2.0 adds an enhanced config file format including “macros”. This overcomes previous limitations regarding chaining WFT commands together that were written to a dynamically generated path. Version 2.0 also includes a number of new command line options, which support features added with this update. Additionally, version 2.0 includes a re-vamped config file that has been better optimized for forensic collection (including more tools). Previous restrictions on verifying cmd.exe before using the tool have been removed to better support people who are using WFT for auditing purposes. While not one of the original design goals, WFT has proven itself quite useful for the auditor and well as the incident responder.

# WFT Usage

- **wft [-h] [-help] [-?] [-usage]**
  - Outputs usage instructions to stdout
- **wft [-md5 filename]**
  - Outputs MD5 checksum of FILE to stdout
- **wft [-fixcfg incfgfile outcfgfile] [-toolpath path\_to\_tools]**
  - Outputs a new config file with updated MD5 checksums
  - Note: Also updates v1.0 config files to the v2.0 format (except <%drive%> macros)

## **wft [-h] [-help] [-?] [-usage]**

Outputs usage instructions to stdout

## **wft [-md5 filename]**

Outputs MD5 checksum of FILE to stdout

## **wft [-fixcfg incfgfile outcfgfile] [-toolpath path\_to\_tools]**

Outputs a new config file with updated MD5 checksums

Note: Also updates v1.0 config files to the v2.0 format (except <%drive%> macros)

The last example of WFT usage ‘-fixcfg’ was added in version 2.0. This option is designed to fulfill two needs:

- 1) Updating the MD5 checksums of all tools listed in the config file
- 2) Update previous config files from v1.0 format to v2.0 format.

Note: While I have made every effort to perform config file updates in an accurate manner, it is impossible for me to account for all possible variants of v1.0 config files. You need to verify that things work as intended in the new file.



# WFT Usage, Continued

- **wft [-cfg cfgfile] [-drive drive\_letters] [-toolpath path\_to\_tools] [-dst destination] [-shell cmdshell] [-noslow] [-nowrite] [-noreport]**
  - Executes WFT as defined in notes

## **-cfg cfgfile**

Uses cfgfile to determine which tools to run (defaults to 'wft.cfg')

## **-drive drive\_letters**

Specifies the drives to be used by wft (defaults to 'C')

## **-toolpath path\_to\_tools**

Defines the path where wft tools are stored (defaults to '.')

## **-dst destination**

Defines the path that reports will be written to (defaults to '.')

Note: Destination can include macros \$magic\$, \$systemname\$, \$date\$, or \$time\$

## **-shell cmdshell**

Redefines shell references from cmd.exe to cmdshell

## **-noslow**

Causes WFT not to run slow (S) executables in cfgfile

## **-nowrite**

Causes WFT not to run executables that write (W) to source machine

## **-noreport**

Causes WFT not to create HTML (H) reports

# WFT Configuration File

- The power of WFT is its config file
- Defines what commands are run, how they are run, and the order they are run in
- WFT collects what the config file tells it to
- Enforces sound forensics (checksums, logging, known trusted binaries, etc.)
- Highly customizable and extendable by the user to allow for specialized responses or it can be used "as is" for a more generic one

This is the config file format used by WFT 2.0:

**ACTION EXECUTABLE MD5CHECKSUM COMMAND OUTPUT MENU DESCRIPTION**

Note: Each of these items is separated by a TAB (white space will not work).

Note: Lines beginning with # are treated as comments.

**ACTION** tells Windows Forensic Toolchest (WFT) how to process each line:

- V** Perform MD5 verification of EXECUTABLE.
- E** Build a COMMAND to execute.
- N** COMMAND produces NO output to md5.
- H** Build a HTML report.
- M** Add a menu heading.
- S** Skip COMMAND if -noslow option is used.
- W** Skip COMMAND if -nowrite option is used.

Note: Multiple ACTIONS can be combined on a line

**EXECUTABLE** tells WFT what Executable this line will be using.

**MD5CHECKSUM** is the MD5 checksum of EXECUTABLE.

**COMMAND** tells WFT how to build the command line to be invoked.

**OUTPUT** is the filename (no extension) to be used for the raw report.

**MENU** sets the text to be used in the Report link or Menu header.

**DESCRIPTION** describes the EXECUTABLE and its purpose.

# WFT Macro Substitutions

- Version 2.0 adds new macro expansions for COMMANDs specified at run time via the command line, via the WFT config file, or from the system properties
  - This overcomes the previous limitation of not being able to chain commands
  - It adds new power to WFT's config file and command line options including allowing for dynamic drive letter expansions

## WFT 2.0 Macros:

- <%executable%> -- the EXECUTABLE specified in the config file
- <%output%> -- the value OUTPUT + '.txt' as specified in the config file
- <%toolpath%> -- the -toolpath directory specified at run time (defaults to '.')
- <%dst%> -- the -dst directory specified at run time (defaults to '.')
- <%cfg%> -- the -cfg config file specified at run time (defaults to '.\wft.cfg')
- <%shell%> -- the -shell specified at run time (defaults to 'cmd.exe')
- <%drive%> -- which is an expanding macro and requires further explanation below

In addition to COMMANDs, these macros also work on the -dst arguments using '\$' notation to replace the '<%>' and '%>' such as \$magic\$, \$systemname\$, \$date\$, or \$time\$

- <%magic%> -- expands to '<%systemname%><%date%><%time%>' and is done first
- <%systemname%> -- system name of the computer for the current run
- <%date%> -- date of the current run in the format 'MM\_DD\_YY'
- <%time%> -- time of the current run in the format 'HH\_MM\_SS'

WFT 2.0 adds a new “macro expansion” option when the <%drive%> tag is used on a line  
The -drive argument should be a list of drive letters to iterate through on commands

Note: -drives defaults to 'C' unless specified at run time

Each line that has a <%drive%> tag will iterate for each drive in the -drives argument

Note: COMMAND, OUTPUT, and MENU must all have this tag if it is being used or else output may be overwritten (this is enforced via WFT for safety)

# How to Use WFT in Practice

- Should be run from CD (or memory stick)
- Requires some up-front setup
  - All binaries (executables and DLLs) being run need to be copied to the CD / memory stick
  - Config file may need to be customized with appropriate tools and MD5 checksums
  - Hint: You can have more than one config file
- Reports should never be written to the target
  - Write to a remote computer via UNC sharename
  - Write to a USB memory stick

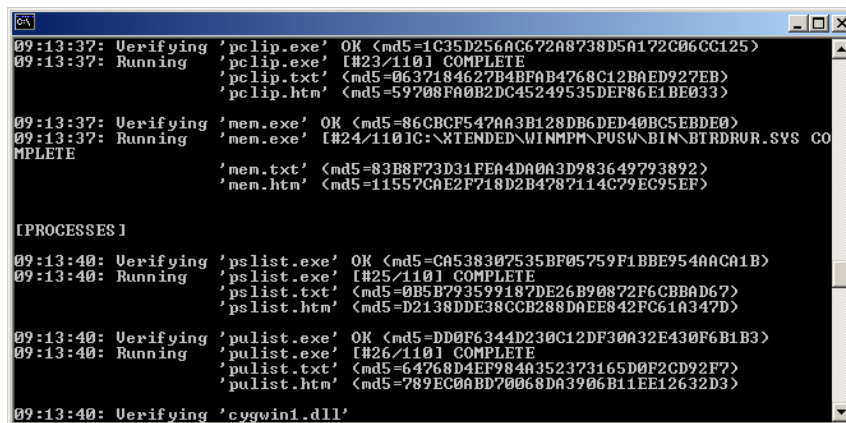
WFT should be run from a CD (or USB memory stick) to ensure the forensic integrity of the evidence it collects. In addition to the WFT binary, users will also need to copy any external programs it will be invoking to the CD / memory stick. The CD or memory stick media should also include a trusted cmd.exe matching the version of the one on the target system to ensure that WFT is being used in a forensically sound manner. WFT version 2.0 removes the requirement for this validation in order to make the tool more useable in an auditing environment.

The config file that is being used to invoke WFT should contain the MD5 checksums of not only all the tools being accessed, but also any external files they require (i.e. any DLLs, config files, etc.). Each of these files should be verified (using the V action in the config file) at least once during WFT execution to ensure that the MD5 is valid. All verifications are logged as part of WFT's execution.

Hint: It is quite possible that as a user of WFT, you may want to build multiple config files for use depending on the type of response desired. Config file can be selected at run-time via the `-cfg` argument.

Output of WFT should usually not be written to the target machine's fixed disk as this would alter the system during the data collection process.

# WFT In Action



```
09:13:37: Verifying 'pclip.exe' OK <md5=1C35D256AC672A8738D5A172C06CC125>
09:13:37: Running 'pclip.exe' [#23/110] COMPLETE
'pclip.txt' <md5=0637184627B4BFAB4768C12BAED927EB>
'pclip.htm' <md5=59708FA0B2DC45249535DEF86E1BE033>

09:13:37: Verifying 'mem.exe' OK <md5=86CBCF547AA3B128DB6DED40BC5EBDE0>
09:13:37: Running 'mem.exe' [#24/110] COMPLETE
'mem.txt' <md5=83B8F73D31FEA4DA0A3D983649793892>
'mem.htm' <md5=11557CAE2F718D2B4787114C79EC95EF>

[PROCESSES ]

09:13:40: Verifying 'pslist.exe' OK <md5=CA538307535BF05759F1BBE954AAC1B>
09:13:40: Running 'pslist.exe' [#25/110] COMPLETE
'pslist.txt' <md5=0B5B793599187DE26B90872F6CBBAD67>
'pslist.htm' <md5=D2138DDE38CCB288DAEE842FC61A347D>

09:13:40: Verifying 'pulist.exe' OK <md5=DD0F6344D230C12DF30A32E430F6B1B3>
09:13:40: Running 'pulist.exe' [#26/110] COMPLETE
'pulist.txt' <md5=64768D4EF984A352373165D0F2CD92F7>
'pulist.htm' <md5=789EC0ABD70068DA3906B11EE12632D3>

09:13:40: Verifying 'cygwin1.dll'
```

This shows a capture of Windows Forensic Toolchest (WFT) in action. In this case, the output. Version 2.0 makes changes to the way this output is displayed making it more compact and easier to read. Additionally this screen capture demonstrates one of the new features in 2.0 where WFT now displays the number of current command being executed along with the total count to be executed.

Note: The number of commands may be greater than the number of lines in the file as the <%drive%> macro is expanded for each of the system drives.

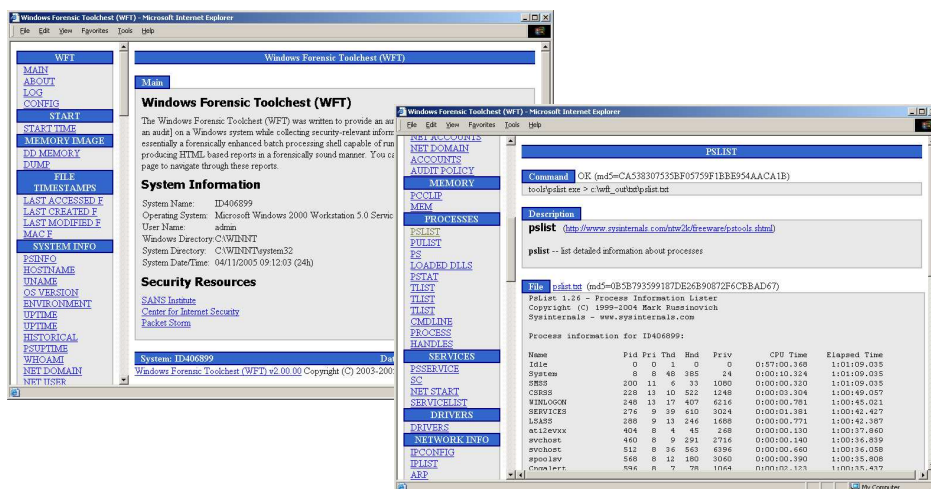
For example, if [-drive argument] was 'CEF' and the COMMAND, OUTPUT, and MENU were:

```
<%toolpath%><%executable%> /C dir <%drive%>:\*.*          <%drive%>_dir  DIR
<%drive%>
```

Then this would expand to three normal entries as:

```
<%toolpath%><%executable%> /C dir C:\*.*  C_dir          DIR C
<%toolpath%><%executable%> /C dir E:\*.*  E_dir          DIR E
<%toolpath%><%executable%> /C dir F:\*.*  F_dir          DIR F
```

# Example WFT Reports



Live Forensics on a Windows System -- © 2003-2006 Monty McDougal

22

Windows Forensic Toolchest (WFT) provides output in two data formats:

**HTML Output:** Opening the index.htm file produced by WFT provides an easy to read and easy to navigate interface to the output of the various tools invoked via WFT. Each of the reports produced under WFT includes the MD5 checksum for the binary being run, the exact command line issued to generate the output, a description of the tool, and the output produced by the tool along with the MD5 checksum associated with the output. The HTML reports are designed to be self-documenting via the text provided in the configuration file.

**Raw Text Output:** This format allows the viewer to see the output of the individual command exactly as it was produced. It is generally a bad idea to, in any way, manipulate data being used as evidence in a court of law. WFT seeks to preserve the original data while providing a user-friendlier HTML version for viewing. The MD5 checksums produced for each of the output files during collection provides a safeguard to ensure the output can be verified at a later date.

WFT Version 2.0 adds two subdirectories for this output – “html” and “txt”. Additionally it supports system/date/time paths with auto directory creation to better facilitate historical comparisons between WFT runs or systems.

# Thank You For Attending

---

Questions?  
Contact Monty McDougal  
monty@foolmoon.net



The author of this tool is open for suggestions, criticisms of this tool, or offers to help improve the tool's config file and/or its documentation. Comments relating to WFT can be sent to the author at [wft@foolmoon.net](mailto:wft@foolmoon.net).

About the author: Monty holds the following major degrees and certifications: BBA in Computer Science / Management (double major) from Angelo State University, MS in Network Security from Capitol College, CISSP, ISSEP, ISSAP, GIAC Certified Incident Handler (GCIH), GIAC Certified Forensic Analyst (GCFA), and serves on the SANS GCIH and GCFA Advisory Boards.

Windows Forensic Toolchest (WFT) and this presentation are  
Copyright © 2003-2006 Monty McDougal. All rights reserved.