# TZWorks<sup>®</sup> Graphical Engine for NTFS Analysis (*gena*) Users Guide



#### Abstract

**gena** couples a number of TZWorks tools into one to aid in NTFS data analysis. This includes **ntfswalk**, **ntfscopy**, **ntfsdir** and **wisp**. The GUI front end integrates the available options into a manageable set of choices for the user. Included in this is the ability to export any data stream that is associated with a file. **gena** can operate on a live volume, an image of a volume or a *VMWare* volume. Summary data can be outputted in one of three parsable formats for easy inclusion with other forensics artifacts. **gena** runs on Windows, Linux and macOS.

Copyright © TZWorks LLC <u>www.tzworks.com</u> Contact Info: <u>info@tzworks.com</u> Document applies to v0.60 of **gena** Updated: Apr 25, 2025

# Table of Contents

1	Introdu	ction3
The	e Layout o	of gena4
2	Source,	Results File, Data/Time Options6
2	2.1 So	urce Options6
	2.1.1	Examining a Mounted Volume or Drive Number6
	2.1.2	Examining a VMWare Volume6
	2.1.3	Examining a Volume Shadow8
2	2.2 Fo	rmat options10
	2.2.1	Results file format
	2.2.2	Date format
	2.2.3	Time format11
	2.2.4	Base-10 format12
3	ntfswal	<i>k</i> dialog tab12
Э	8.1 Fil <sup>-</sup>	tering Options – ntfswalk
	3.1.1	Filtering on NTFS File Record data (external data)14
	3.1.2	Filter on file content data (internal data)15
	3.1.3	Selecting what clusters to scan and whether we are just interested in deleted records15
	3.1.4	Selecting Files in a Time Range16
	3.1.5	Miscellaneous Results file options16
	3.1.6	Extraction Options16
	3.1.7	Manually adding and deleting filters18
4	MFT re	cord metadata tab20
5	Volume	tab21
6	wisp ta	b22
7	Availab	le data for selected MFT record23
8	Utilities	Menu24
9	Use-Ca	ses

9.1	General Analysis on Targeted files	. 25
9.2	Incident Response Collection	. 25
9.3	Generating Hash Sets on Targeted File types	. 26
9.4	Support of Collection into Timeline Analysis	. 27
10	X-Window Dependencies	. 27
11	Authentication and the License File	. 27
11.1	Limited versus Demo versus Full in the tool's Output Banner	. 27
12	References	. 29

# TZWorks<sup>®</sup> NTFS Analyzer/Viewer (*gena*) Users Guide

Copyright © *TZWorks LLC* Webpage: http://www.tzworks.com/prototype\_page.php?proto\_id=28 Contact Information: <u>info@tzworks.com</u>

## **1** Introduction

The forensic community has been providing consistent feedback on the TZWorks toolset in identifying which tools and capabilities are important to them. This feedback is valuable and allows us at TZWorks to focus on areas useful to the community and is one of the reasons we continue to develop new tools. One item we hear repeatedly is a request to provide a GUI (Graphical User Interface) front end to some of the TZWorks command line tools. While we internally prefer command line tools for automated processing, we do have a handful of GUI based tools that we develop for internal use only, primarily for reversing and analyzing new artifacts. So we decided to take one of our internal tools, take out some of the more arcane options, and merge the backend processing with *ntfswalk* and some other tools to come up with *gena*.

The name *gena* is short for *Graphical Engine for NTFS Analysis*. Along the way, we made some significant additions to *ntfswalk*, as well, to allow **gena** to be used as a flexible tool for data extraction. We also incorporated some capabilities from *ntfscopy ntfsdir*, and *wisp* into *gena*. So while this document is a *gena* user's guide, there will be references to these other tools throughout.

When developing a GUI app we have stuck with the FOX (Free Objects for X) C++ toolkit, since it is cross platform, light weight, fast in performance and compiles into a static library that is relatively small when compared to the other GUI toolkits. FOX, however, has some drawbacks in that it may not have the look and feel that a normal Windows application may have. The GUI front end for *gena* uses the FOX toolkit.

Similar to the other TZWorks tools that were mentioned, *gena* is designed to work with live (mounted) NTFS volumes. There is also functionality for traversing either NTFS images (a) created with the *dd* utility or (b) from a monolithic volume consisting of *VMWare* VMDK files. Whether *gena* is being used for live incident response collection or to process an image in an off-line manner, there are options to filter on file extensions, a timestamp range, various binary signatures, partial filenames, and directory contents. For targeted files found, one can list the summary metadata, extract the header bytes of the file data, or extract the entire file contents into a designated directory. Since the GUI front end and backend parsing engine are both Windows API agnostic, there are compiled versions for Windows, Linux and macOS.

## The Layout of gena

The *gena* GUI is divided into two main window panes. The first pane is on the left, and is a quasi-Windows Explorer view of the volume under analysis. The right pane is governed by a series of tabs that reflect differing functionality. The number of tabs is dependent on the MFT record (or *inode*) that is selected on the left tree-view pane. This is the area where one can invoke the *ntfswalk* tool or peer into the details of an MFT record. The first and default tab is for the *ntfswalk* dialog. It exposes various options unique to *ntfswalk* and will be discussed in a later section.

When one loads a volume for analysis, a series of additional tabs automatically become visible. This includes: a tab for MFT record metadata (timestamp, cluster runs, etc.), a tab for the hexadecimal dump of the file record under analysis, and a separate tab for each NTFS attribute containing data. The term data defined here, can include: (a) the unnamed data stream, (b) any alternate named data streams, (c) the directory's INDX data, (d) a Logged Stream data, and (e) various others. To keep things simple, each unique data set is selectable with a separate data tab.

Below is a screenshot of analyzing a 'C' partition from a local machine while the *\$MFT* file is selected. When looking at the directory tree, one can see any alternate data streams via the superimposed red letter 'A' on the file icon. This is only an artifact of **gena** to flag to the user that an alternate data stream is present. Another noteworthy icon that is displayed is for reparse points that are found that form junctions to other directories. These types of reparse points are shown by superimposing the red letter 'S' on the file icon ('S' is for softlink or symbolic link). In the screenshot, the volume under analysis was a Windows 7 partition, so the "C:\Documents and Settings" is a softlink to the "C:\Users" directory.

For the *\$MFT* record selected, *gena* shows five tabs on the right pane. The first three are always present when a file is selected. The last two are unique to whether *gena* thinks there are data associated with the selected *inode* to be displayed, and this in this case, creates a tab for the 'normal data' and 'bitmap' data. The 'normal data' is defined here to mean the 'unnamed' data stream.

Since the second (metadata) tab is selected in the example, one can see the various NTFS file record attributes in an interpreted manner associated with this inode. This includes standard attribute properties/timestamps, filename timestamps, and data attribute cluster runs. If raw data is preferred over the interpreted data, one can select the third tab, which shows the raw volume data of the specific *inode* (MFT entry) selected.

Eile Data Format Utils		Help
C  SMFT (0) SMFTMer (1) SLogFie (2) SVolume (3) SAttrOef (4) SBitmap (6)	Path: [root]\\$MFT File Record: MFT entry: 0x00000000 [0] Seq Num: 0x0001 [1] Type: file Ref Count: 1 Standard Info Attrib: (HIDDEN, SYSTEM)	
SBoot (7)     SBadClus (8)     SBadClus (8)     SSecure (9)     SUpCase (10)     SExtend (11)	<pre>modified: 09/12/2014 00:19:19.533 accessed: 09/12/2014 00:19:19.533 mft mod: 09/12/2014 00:19:19.533 created: 09/12/2014 00:19:19.533 secure id: 0x00000100 [256] quota id: 0x00000100 [0] Jrnl num: 0x00000000 [0]</pre>	
SRecycle Bin (57)     SWindows -BT (84487)     Det (60869)     Documents and Settings (     dump (65332)     Imtel (59779)	<pre>Filename Attrib: \$MFT parnt mft: 0x00000005 [5] parnt seq# 0x0005 [5] modified: 09/12/2014 00:19:19.533 accessed: 09/12/2014 00:19:19.533 mft mod: 09/12/2014 00:19:19.533 created: 09/12/2014 00:19:19.533</pre>	
MSOCache (128589)     mtddk (84437)     PerfLogs (58)     Program Files (60)	Data Attrib: (uppened) allocated x000017100000 file size xx000017100000 ntfswak metadata voidata normal bimap	

To look at the cluster runs for any data attribute, one need only select the appropriate data tab. The data tabs, have a number of options available, such as: (a) jumping to a specific offset in the cluster run, (b) finding a string or pattern, (c) exporting the data section to a separate file, (d) showing the data as hex, ASCII, or Unicode. (See the screen shot below for an example).

The Tank shipe Tank				Teb
c SMFT (0) SMFTMrr (1) SLogFit (2) SVolume (3)	<ul> <li>Size stats</li> <li>Alloc size</li> <li>Vald size</li> <li>Slack size</li> </ul>	0x17100000 0x17100000 0x00000000	Data options (offset/search/export) Input: 0x00000000 Goto FindString FindNext Export	Show data as . (* Hex C Asci C WideAsci C UTF-16 Start * End
SAmper (4)     Samp (6)     Searce (7)     Searce (9)     Secure (9)     Supcase (10)     Secure (9)     Supcase (10)     Secure (11)     Secure (11)     Secure (11)     Secure (11)     Secure (11)     Secure (12)     Del (60869)     Documents and Settings     dump (65332)     MSOCache (128589)     MSOCache (128589)     MSOCache (128589)     MSOCache (128589)     MSOCache (58)	0000 0000 0000 0010 0000 0110 0000 0110 0000 0110 0000 0110 0000 0110	46         49         42         45           11         89         81         61         69           38         81         61         61         69           38         81         61         61         69           38         81         61         61         60           23         60         12         32         32         60	38         00         01         00         04         04         04         02         04         00         00         00           38         00         01         00 </td <td>FILER</td>	FILER

## 2 Source, Results File, Data/Time Options

#### 2.1 Source Options

Currently, there are five types of source data **gena** can analyze, including: (a) NTFS drive/volume stored as a 'dd' type image, (b) NTFS mounted drive/volume, (c) an exported \$MFT file, (d) a VMWare volume, or (e) a Volume Shadow Copy. To analyze an image or mounted volume, one uses the "File" menu as shown below:

Eile Data Format Utils	
Open NTFS Image File	Ctl-O
Open NTFS Mounted Drive/	Volume Ctl-M
Open Exported SMFT file	Ctl-E
Open VMWare Volume	Ctl-V
Open Volume Shadow	Ctl-S
Quit	CtI-Q

#### 2.1.1 Examining a Mounted Volume or Drive Number

By selecting the second option in the File Open options, one can select either a mounted volume letter or a mounted drive number. The following dialog is displayed after the initial menu selection:

pen Data Source	Contractory of Contra	
Source		
Option	Mounted Volume Letter	
	Mounted Volume Letter	
	Mounted Drive Number	

Selecting a mounted volume letter is straight forward; however, selecting a drive number will require one to specify an offset of the volume on the drive. After selecting a drive number, *gena* will read the partition tables and display available offsets for the selected drive.

Physical Drive	Drive#0	
Option	0x000000100000	
	0x000000100000	

#### 2.1.2 Examining a VMWare Volume

By selecting the VMWare option in the File Open options, one can select the desired VMDK snapshot that is desired without worrying about its dependencies. For example, selecting the 5<sup>th</sup> snapshot as shown below, causes **gena** to look at any dependencies available.

Name 🗸	Туре	Size	Modified Date -
Win7 home prem-000001.vmdk	VMDK File	773193728	07/12/2014 12:30:52
Win7 home prem-000002.vmdk	VMDK File	3563782	08/15/2014 17:04:24
Win7 home prem-000003.vmdk	VMDK File	649330688	07/12/2014 19:37:44
Win7 home prem-000005.vmdk	VMDK File	941555712	08/29/2014 13:14:17
☐ Win7 home prem.vmdk ↓	VMDK File	1662274	07/12/2014 12:23:41

This is the response gena has (see below). It knows the *Win7 home prem-000005.vmdk* depends on *Win7 home prem-00002.vmdk*, which depends on *Win7 home prem.vmdk*, alerts the user and then tries to open the volume with these 3 VMDK files.

٢	Need to select dependent vmdk files: => vmdk file: V:\vmware\Win7 home prem-000005.vmdk depends on parent vmdk "win7 home prem-000002.vmdk" => vmdk file: V:\vmware\win7 home prem-000002.vmdk depends on parent vmdk "win7 home prem.vmdk" gena will try to automatically include these vmdk's
	gena will try to automatically include these vmdk's

E ifie: V:/vmware/Win7 home prem-000005.vmdk	4	Path: [root]\\$MFT
5MFT (0)		
- SMFTMirr (1)		File Record:
SLooFile (2)		Sea Num: 0x00000000 [0]
SVolume (3)		Type: file
D Sott Def (4)		Ref Count: 1
D SBitman (B)		and the second second second second
D shart (7)		Standard Info Attrib: (HIDDEN, SYSTEM)
Boot (/)		modified: 01/10/2010 05:50:20.812
SBadClus (8)		accessed: 01/10/2010 05:50:20.012
A SBadClus SBad (8)		scanted: 01/10/2010 05:50:20.012
SSecure (9)		created: 01/10/2010 05:50:20.012
A SEAMURESOE (0)		quota id: 0x00000000 [230]
B successors (a)		Jrn1 num: 0x00000000 [0]
SUpcase (10)		(1), (1), (1), (1), (1), (1), (1), (1),
Extend (11)		Filename Attrib: \$MFT
🕂 🦲 SRecycle.Bin (57)		parnt mft: 0x00000005 [5]
+ C Boot (41244)		parnt seq# 0x0005 [5]
(1) Confin Mei (42726)		modified: 01/10/2010 05:50:26.812
Conng.Mar (42/20)		accessed: 01/10/2010 05:50:26.812
Documents and Settings (9363) -> C:\Users		mft mod: 01/10/2010 05:50:26.812
i 🦲 dump (46224)		
(# Cal history (28875)	-	ntiswam metadata voi data normai data bitmap

#### 2.1.3 Examining a Volume Shadow

Volume Shadow copies, as is discussed here, only applies to Windows Vista, Win7, Win8 and beyond. It does not apply to Windows XP.

When selecting Volume Shadow as the source, one will be asked which Volume Shadow Copy to analyze. Each Volume Shadow has an index. *gena* will automatically enumerate all the available indexes and display then in the dialog box below. This index equates to the *HarddiskvolumeShadowCopy#*, where the *#* is the index number.

Index	1	
	1	
	2	

After selecting the index, one is presented with a populated tree view of the NTFS volume represented by the Volume Shadow. *gena* annotates the root as the symbolic link to the Volume Shadow that was selected.

S WNGLOBALROOT/Device/Harddiak/Volume5hadowCopy	3 Path: [root]\\$HFT
5MFT (0)	
D SHETTHER (1)	File Record:
D BLOGFER (2)	Fri entry: 0x00000000 (0)
D EValume (3)	Type: file
D samper (a)	Ref Count: 1
D Statman (E)	
D Short (7)	standard Invo Attrib: (HIDDEN, SYSTER)
D Shadhar (D)	accessed: 07/18/2013 17:28:00.468
A That the find (T)	mft mod: 07/18/2013 17:28:00.468
A 3080-148 3080 (0)	created: 07/18/2013 17:28:00.468
Secure (9)	secure id: 0x00000100 [256]
A Secure SSDS (9)	quota 1d: 0x000000000 [0]
SupCase (12)	Star next exceeded [4]
3 🚍 \$Extend (11)	Filename Attrib: \$MFT
🖹 🦲 SRecycle.Bin (57)	parnt mft: 0x00000005 [5]
🛞 🦲 Dell (59635)	parnt seq# 0x0005 [5]
Documents and Settings (13763) -> CNUsers	MODITIES: 07/18/2013 17:28:00.408
🗄 🧰 dump (76830)	mft mod: 07/18/2013 17:28:00.468
+ 🛄 Intel (60235)	created: 07/18/2013 17:28:00.468
+ (i) misc (60722)	
+ 3 MSOCache (65202)	Data Attrib: (unnamed)
Perfl.pgs (58)	file size: 0x00000e040000
Program Files (60)	valid data: 0x000000e040000
10 Deveran Files (x86) (347)	cluster runs
Beneratinta (153)	0x000c0088 -> 000cbeff
in a regularization (2002)	ex08160403 0x80161562
macovery (1909)	
System Volume Information (15974)	<ul> <li>Ittewak metadata vol data normal data bitmap</li> </ul>

From here, one can analyze the metadata for any folder or file, which is shown on the right pane. Using the tabs on the right pane, one can analyze the raw data metadata for the file record that comprises the MFT entry or any of the data streams included in that MFT entry. The tab called "normal data" is for the default 'unnamed' data stream. For those MFT entries that have many attributes backed by data, gena dynamically adds additional tabs for each one.

If that entry is selected, a tab called "wisp" will also show up that will identify all the INDX records in the MFT entry. For example, when the \$Recycle.Bin is selected, the "wisp" tab is created and shows all the child entries that this folder is a parent of. For the wisp tab, one can select to see just the valid children (valid means, not deleted) or the slack entries (which are the children entries that have been deleted) or both, which is selected below.



## 2.2 Format options

There are a few data format options, grouped by the following categories: (a) Final output, (b) Date Format, and (c) Time Resolution.

#### 2.2.1 Results file format

The first category, final output, tells **gena** how to format the results file **ntfswalk** produces. This can either be: (a) default output, (b) CSV format, (c) *Log2Timeline* format, (d) *BodyFile* format, (e) a *Hashfile*, or (f) a CSV format where there is only one path/file entry per line. See the menu screen shot below:



The default option uses one line to specify the attributes per inode and uses the pipe character '|' as a separator. The CSV option is similar to the default option, but forces all output to be in comma separated value format. Thus, any data that had commas in its name are changed to space character to protect the integrity for the CSV fields.

The *Log2Timeline* option forces all output to be in CSV format and tries to conform to the log2timeline format specified by the website <u>http://log2timeline.net/</u>. This output needs to be independently verified for correctness.

The *Bodyfile* option outputs the data fields in accordance with the *body-file* version3 specified in the *Sleuthkit*. The date and timestamp outputted to the *body-file* are in terms of UTC. So if using the *body-file* in conjunction with the *mactime.pl* utility, one needs to set the environment variable TZ=UTC. This output also needs to be independently verified for correctness.

The *Hashfile* format will generate both an MD5 and an SHA1 entry as well as other metadata per *inode* that was processed.

The last option is the *CSVPerPath*, which outputs only one unique path/file entry per line. The fields are formatted as CSV.

Note: If extracting files during the run, only the last option is available. If not extracting file content data, then all five options are available.

#### 2.2.2 Date format

The date format can be selected based on desired convention. The Date Format menu offers three common format options: mm/dd/yyyy, dd/mm/yyyy or yyyy/mm/dd. The default is set to the USA convention of: mm/dd/yyyy. The graphic below shows these options:



#### 2.2.3 Time format

The time format has three resolution options: (a) seconds, (b) milliseconds and (c) microseconds. The default is set to milliseconds. The graphic below shows these options:



#### 2.2.4 Base-10 format

The final format option is to select how you wish size and offset numbers to be represented (base 10 or base 16). This option was not put into the menu, but put into the *ntfswalk* dialog tab. Setting this option will reflect changes throughout the other tabs as well. The screenshot of this option is highlighted below. The default is unchecked or hexadecimal.

~	10		
<u> </u>	Find namelext (faster)	Find Signatures (slower)	UTC-Range (mm/dd/yy hh.mm.ss)
- 📑 SMFT (0)	Prefetch Files	LNK Files	Start
- D SMFTMirr (1)	LINK Files	Event Loos	Start 100/00/0000 00:00:00
SLogFile (2)	Lumol ists	E Registry Haves	Stop: 01/01/2100 00:00:00
- SVolume (3)	E index dat	C SOI de Databases	1
SAttrDef (4)	Thumblisis	F Event be	Results File Misc Options
S8tmap (6)	C Office/DDF Files	Chose and	Use Base 10 vice Hexadecimal
- 🗎 \$Boot (7)	Email Flag	T All of the shows	Record volume offset in output
- StadClus (8)	Cranhice Files	Parent and debute	Record cluster run
A Shad Chie Shad (8)	- Graphics files	Scan where & type records	

## 3 *ntfswalk* dialog tab

*gena* was originally designed to be a front end to the *ntfswalk* command line tool. The additional capabilities added to version 0.45 of *ntfswalk* coupled with **gena** make it easy to target specific files in a scripted manner, whether for incident response or normal forensic data collection. *gena* puts many (but not all) of the available *ntfswalk* command line options at easy access to the user and graphically arranges them by functionality. Therefore, this is the tab to use, if you want to extract many files or analyze an entire volume.

If using *gena* just to setup a script for *ntfswalk* to run later on another target box, one can use *gena* to select the desired options without loading an image, and select the "Generate Script" button. This action will package all the options selected, generate a script that is compatible to be invoked directly by *ntfswalk* (in a command line mode). To handle this script, a new *-script* command line option has been added to *ntfswalk*. See the *ntfswalk* documentation for more details.

On the other hand, if one wants to perform a data collection or volume analysis on a live system (or offline image), one can point *gena* at any NTFS volume (or image) and it will operate on that target. Selecting the "Spawn *ntfswalk*" button will do just that: spawn an instance of *ntfswalk* as a separate

process passing to it the options the user has selected. **gena** and **ntfswalk** really do not care whether the target volume is a live mounted image or a 'dd' image. They process them the same way. As an aside, since **gena** will spawn **ntfswalk**, both tools should be in the same directory along with their authenticating licenses for this mode to work. Using this latter technique, **gena** can spawn as many instances of **ntfswalk** that is desired while they run in parallel, assuming the host machine can handle the processing and memory load.

Since there is this close relationship between **gena** and **ntfswalk**, only version 0.45 of **ntfswalk** and later work with **gena**. Below is a picture of the location of the buttons used to either generate a script or spawn **ntfswalk**.



## 3.1 Filtering Options – ntfswalk

The filtering options are where one can hone exactly what type of files they are interested in. This turns out to be the primary way to speed up any target collection. To understand the filter options, some background on *ntfswalk* is useful. Version 0.45 of *ntfswalk* has been enhanced in a number of ways from the prior versions. While still evolving, filtering is one of the bigger enhancements and was geared to help out in incident response collection. Below is a diagram to showing some of the major filtering options.



The filtering architecture has been beefed up to allow for more combinations of options, some of which use OR logic and others which use AND logic. Consequently, as the number of command line options increased, the complexity (and confusion factor) increased as well. Fortunately, the *ntfswalk* dialog tab in *gena* keeps the OR and the AND logic straight between the various options by the various grouping of options.

From the above diagram, one can filter on any number of extensions, partial names, directories, and signatures. This means they will all use OR logic, which translates to, if any of the *inode's* passes any one filter mentioned above it will go to the next step. The AND logic comes into play when looking at (a) filtering by date range and (b) filtering on deleted files. Once an *inode* passes the filtering tests, it then proceeds to be outputted to the results file, and if, instructed, copied to the specified directory.

## 3.1.1 Filtering on NTFS File Record data (external data)

The most basic (and fastest) filtering one can perform is by filtering on the NTFS file record (or external) metadata. This includes things as: (a) file name, (b) file extension, (c) time stamp, (d) parent directory, (e) or *inode*. The other mode of filtering is on internal, file content data. This mode of filtering is discussed in the next section.

All external filtering options are shown by the checkboxes grouped via the "Find name/ext (faster)" title and shown below. One can select none, one or more options to filter on.



## 3.1.2 Filter on file content data (internal data)

The more thorough (and slower) filtering one can perform is by filtering on the file content data. The *ntfswalk* engine will use predeterministic signature analysis to determine whether a file meets one of the available categories. Only a few common ones were added to *gena*. These include: (a) LNK files, (b) Event logs, (c) Registry hives, (d) SQLite3 databases, and (e) executable and library files.



# 3.1.3 Selecting what clusters to scan and whether we are just interested in deleted records

This category of options is broken up into four *use-cases*. The first set is whether one just wishes to scan (a) all records in the \$MFT data or (b) just the deleted records in the \$MFT data. The second set is whether one wants to scan (c) all the clusters of a volume or (d) just the deleted clusters of a volume. A fast option is to only scan the \$MFT data (eg. data in inode 0), while the more complete option is to scan "All clusters/All records". The default option is the former (all records in the \$MFT data section).



#### 3.1.4 Selecting Files in a Time Range

One can select a date range using the entry box shown below. If at least one date (of the 4 MACB standard information dates) for an inode is within this range, it will pass the test and be processed. Note all dates need to be expressed in UTC format and the date string needs to conform to: mm/dd/yyyy hh:mm:ss.



#### 3.1.5 Miscellaneous Results file options

Each run will produce a results file. In addition to the normal data that is provided, other data can be inserted as well. In addition to changing the default hexadecimal output to base10 discussed previously, one can record the volume offset of the file record as well as record the cluster run. These options are shown here.



#### **3.1.6 Extraction Options**

If desiring to extract files to an archive directory, one can select the "Copy Target Files" checkbox. If one is worried about copying malicious data from a target box one can change the extension of the file extracted by appending a \*.bin to the file, so it is not accidently run should the file contain some malicious executable content. To do this, select the "Append .bin as extension" checkbox.

Other options include copying the file as 'raw' which means copy the file as the cluster data is represented and don't translate the data (this applies to the files that use the native NTFS compression). Also the 'raw' option will copy all allocated clusters associated with the file and not just the 'used'

clusters. Therefore the 'raw' option will yield the slack that is available. The last option is whether one wishes to include any sparse clusters as part of the file copy. The sparse clusters, while present on some files, are not backed by physical clusters and therefore **ntfswalk** will copy zeros in place of these sparse clusters, if this option is selected.



The extraction directory for both the results file and any files extracted is specified by the text entry box below. The default directory is "GENA\_Results" under the current working directory. Whatever directory is inputted here will be used as the parent directory for archiving data. *ntfswalk* will create subdirectories under this parent directory to categorizes different runs as well as categorize data within a run.

Results	Directory -
.\GENA	Results

Below is an example of selecting all external filter checkboxes and extracting the resulting data. Notice the subdirectories that are created under the parent directory *GENA\_Results*. The subdirectory hierarchy is as follows: (1) name of the volume being scanned, (2) timestamp of run with the tool name appended, (3) subcategories of deleted, valid, etc. to denote whether the extracted file is a deleted or normal (in use) file, (4) subcategories of whether the file came from a users' local directory or whether it was from a location not specified from a user account (called non\_acct), and (5) a folder break out of the filter used to target the file.



For each file extracted, the file name will consist of 3 items: (a) the timestamp of the last modify time, (b) the computed MD5 hash of the file, and (c) the original filename. See the results of the lnk files extracted from the admin account below. Should one wish to change the native extension of the file during the copy, one can use the "*Append .bin as extension*" checkbox.



#### 3.1.7 Manually adding and deleting filters

*gena* automatically lists any filters selected (minus the date range) in a list box. Thus, any selection made from the filter checkboxes will show up in the list box. Below is a sample of selected filters and how these are annotated in the list box. The purpose of the list box is to allow a user to fine tune what is filtered and what is not.

As an example, look at the snapshot below. In the example, *Prefetch files, LNK files, Jump Lists* and *\$Recycle.Bin* are selected from the first set of filters. Looking at the list box, these show up as \*.pf, \*.lnk, \**destinations-ms*\* and *\$Recycle.Bin*\\*? entries, respectively. The first two are filters that will target the extensions \*.pf and \*.lnk. The next item is a filter for a partial name; which is present in both *automatic* and *custom Jump List* type files. The last item is directory entry that will tell **ntfswalk** to pull all the MFT entries from the *\$Recycle.Bin* directory and subdirectories.

Continuing with the same example, other filters are selected in the snapshot as well. In this case, those that are signature based, such as: *event logs, registry hives*, and *SQLite databases* are selected. These get annotated in the list box as signature filters.

Any one of these filters in the list box can be removed either by unchecking the appropriate filter check box or by selecting one or more list box entries and pressing the "Remove Item" button. Likewise, additional filter items can be added to the list; this can be done in one of two ways: (a) using the "Add File" button and/or (b) using the "Add Text Search" button.



The first option is the "Add File(s)" button. This is used when selecting (highlighting) one or more items from the directory tree on the left pane and pressing this button. The selected entries will be added to the list box. The second option is the "Add Text Search" button. This offers the user some additional flexibility by allowing him/her to add a custom filter. This can be either another extension type, partial name to filter, a directory to scan or a path/filename to filter. Below is an example of adding the change log journal file is shown below.

dd Search String	And the other Designation of the Owner, which the Owner,	
String	SExtend\SUsnJml:SJ	
Option	Path/filename	F
	Extension Partial name	
	Path/filename	
	inode Directory Directory inode	

## 4 MFT record metadata tab

The metadata in this tab includes an 'interpreted' version of the NTFS file record. This means that, if **gena** sees a standard information attribute, it will show the properties and timestamps associated with this sort of NTFS attribute. Likewise, if **gena** sees a filename attribute, it shows the name, parent inode and timestamps associated with this attribute. For data attributes, **gena** shows the size data (allocated vice file size) as well as any cluster runs that house the data. For other attributes where parsing is not done, but data is available, **gena** will show these attributes in a separate tab.

Earlier, there was example of opening a Volume Shadow, where the *file record* metadata was shown for the \$MFT. This was done by clicking on the left hand pane of the tree view of the GUI. Another way to display *file record* data if one has the inode number of the MFT entry they wish to analyze, is to right click in the right pane of the GUI and a context menu will be displayed (shown below). This context menu is only available if the user is located in the 'metadata' tab.

3 N/VGLOBALROOT/Device/Harddisk/VolumeShadowCopy	3 Path: [root]\\$MFT
SMFT (0)	
SMFTMer (1)	File Record:
SLogFile (2)	Sed
D SVolume (3)	Typ Show inode
SAttrDef (4)	Ref Dumo attribute data
b sBitmap (6)	Stand
B 58oot (7)	modified: 07/18/2013 17:28:00.468
- D SBadClus (8)	accessed: 07/18/2013 17:28:00.468
A StadCke Stad (8)	mft mod: 07/18/2013 17:28:00.468
D SSecure (B)	created: 07/18/2013 17:28:00,468
A SSecure 8575 (0)	guota id: 0x00000000 (0)
D SiloCase (10)	Jrn1 num: 0x00000000 [0]
E Statest (11)	and the second second second second
Carl Chamaria Rin (CT)	Filename Attrib: SMFT
	parnt seg# 0x0005 [5]
Concerning and Fertimers (17782) - Children	modified: 07/18/2013 17:28:00.468
Documents and Settings (13/63) -> C-Susera	accessed: 07/18/2013 17:28:00.468
annb (seaso)	mft mod: 07/18/2013 17:28:00.468
et et (60235)	created: 0//10/2013 1/:20:00.400
🖲 🛄 misc (60722)	Data Attrib: (unnamed)
H MSOCache (65202)	allocated: 0x000008e040000
PerfLogs (58)	file size: 0x00000e040000
🖹 🦲 Program Files (60)	
Program Files (x86) (247)	0x000c0000 -> 0x000cbeff
🕸 🧱 ProgramData (363)	0x00f6d423 -> 0x00f6f562
Recovery (1969)	
System Volume Information (15974)	. ntfswak metadata voi data normal data bitmap

By selecting "Show inode", a dialog box will pop up allowing one to enter the inode value they wish to retrieve. For example, let's say one wanted to evaluate what file was associated with inode 215435, one could enter this value (or the hex value 0x3498B) into the dialog box and it would retrieve the *file record* associated with that MFT index.

put Inode		
inode:	0x34988	
		Cancel Accept

From the file record (shown below), one can see the file associated with that inode value. Further, one could select any of the other tabs at the bottom and review the raw file record data or the actual data of the file.

File Record:		
MFT entry:	8x0003495b	(215435)
Seg Num:	8x888d [13]	
Type:	file	
Ref Count:	1	
Standard Info	o Attrib: (A	ARCHIVE)
modified:	01/09/2013	19:26:04.000
accessed:	01/09/2013	19:26:04.000
mft modi	08/13/2014	18:23:15.194
created:	01/09/2013	19:26:04.000
secure id:	0x00000460	[1120]
quota id:	8×86666666	[0]
Jonl num:	8x36ded438	[920572976]
Filename Att	rib: sdelete	exe
parnt mft:	8x8881629b	[98779]
parnt seq#	8x004e [78]	
modified:	08/13/2014	18:23:15.153
accessed:	08/13/2014	18:23:15.153
mft mod:	08/13/2014	18:23:15.153
created:	08/13/2014	18:23:15.153
Data Attrib:	(unnamed)	
allocated:	8x0000000	27000
file size:	0x80808883	26858
valid data	: 0x0000000	26058
cluster run Bx01065c	ns c4 -> 0x0100	55cea
Total clust	ters [real]:	0x888888827
Table 1 aller	ters [snars/	1: 0x888888888

## 5 Volume tab

This tab shows a hexadecimal view of the volume under analysis. When synced with the selected inode from the directory tree, it will display the raw file record data, as shown below.

] ¢	Display data based o	n Explicit Offset	VSize	Offset relative to	
- (0) SMFT (0)	( inode selected	Offset.	0xc398e400	C Custer number in volume	1
SMFTMer (1)	C Hannah Inneller	· · · · · · · · · · · · · · · · · · ·		G. University of Tank	
SLogFile (2)	· manyary reports	Display		A CONTRACTOR CONTRACTOR	
- D SVolume (3)	Contract of the second			1	
D SAMPATIAN	0395 e400: 46 49	4c 45 30 00 03 00	ed ed 48 11 e1	00 00 00 FILE0	
D seases (4)	8398 6418: 82 68	01 00 38 00 05 00	be e1 ee ee ee	84 88 88	2.1
S8tmap (6)	0358 e420: 00 00	00 00 00 00 00 00	86 88 88 88 35	e6 00 009	
- 58oot (7)	0398 e430: 96 59	47 11 00 00 00 00	10 00 00 00 60	40 60 60 .YG	÷
Th StadClus (5)	8358 6448: 00 00	00 00 00 00 00 00	45 00 00 00 15	00 00 00H	÷ .
	0390 e450: eb 94	f1 cc d4 #3 ce 01	eb 94 f1 cc d4	\$3 Ct 01	
A StadClus:Stad (8)	0390 e460: eb 94	f1 cc d4 II3 ce @1	eb 94 f1 cc d4	83 Cé 01	
Secure (9)	0398 e470: 26 02	00 00 00 00 00 00	66 59 56 66 66	ee ee ee ä	
Secure \$505 (9)	9358 6458: 00 00	00 00 01 01 00 00	00 00 00 00 00	ee ee ee	
D SileCom (10)	0356 6456: 00 00	ee ee ee ee ee ee	50 00 00 00 70	46 56 66	÷.,
Subcase (10)	3105 42h0- 0h 00	00 00 00 00 01 00 00	ah 64 f1 rr d4	00 01 00	
E SExtend (11)	8398 44/81 ab 94	f1 cc d4 83 c# 81	ab 94 F1 rr 64	#1 re #1	
SQuota (24)	0195 e4d8: eb 94	f1 cc d4 03 ce 01	50 00 00 00 00	00 00 00	
D construction	0355 6460; 00 00	00 00 00 00 00 00	25 00 00 00 00	00 00 00 ······	
	0398 e4f0: 08 80	24 00 55 00 73 00	6e 00 48 00 72	00 5e 00\$.U.s.n.J.r.n	
SReparse (26)	0398 e500: 6c 00	00 00 00 00 00 00	50 50 50 50 50	ee ee ee 1	
Rmf/etadata (27)	0398 e510: 01 02	48 00 00 50 03 00	00 00 00 00 00	00 00 00H	
D Silan Irel (58937)	0396 e520: 2f bf	03 00 00 00 00 00	58 00 84 80 00	00 00 08 /P	÷
	0390 ±530: 00 00	f3 3b 60 60 😏 60	70 a0 f1 3b 00	00 00 00;pj	÷ .
A BOARDING A CONST.	0398 e540: 70 m0	f1 35 88 83 88 88	00 00 23 02 00	00 00 00 p;#	
A SUanJmtSMax (589	8390 6550; 24 00	4a 60 87 19 ff ff	83 88 55 83 42	30 22 63 \$.780"	2
+ SRecycle Bin (57)	8398 £568: 2d 7f	01 00 fB ff ff	58 50 50 60 40	ee ee ee	0
	- A148 +570: AR B4	11 AB 85 88	70 00 00 00 70	AR AR AR	

With this tab, one has the freedom to look at any volume offset that is desired. To do this, just select the "Manually inputted" radio button and the offset field will be activated as well as the display button. One has the option of traversing the volume either by volume offset address or by volume logical cluster number.

## 6 wisp tab

When selecting any directory from the tree view, a *wisp* (Windows INDX Slack Parser) tab is displayed. Since *gena* can parse the INDX attribute to find the children to a directory, we incorporated a lite version of the TZWorks *wisp* engine. This will allow *gena* not only to parse normal children but also slack entries (or children that have been deleted). This is very useful in identifying deleted files after they have been wiped from the NTFS volume, since the file artifact could still be in the parent directory's INDX attribute. This is best shown via an example.

In the screen shot below, the directory named with the users' security identifier (SID) is opened. In the directory, only one file is shown (the desktop.ini file), which means the user cleaned out the trash. When going to the *wisp* tab, and selecting the "Slack" radio button, one can see from the summary text in the right pane window, that *wisp* was able to identify just 79 entries, where all but one was in slack space and can be analyzed. Each entry will have a name, file size, and a set of timestamps. For additional verification, we added the ability for each entry to show the source data (or hex dump of the data that was parsed), complete with actual offset of where the data resides in the INDX cluster run.

c	Which entries to parse. C Vald Slack C Both	Misc options	Export	wisp Windows INDX Slack Parser Lite version. Download full version from T2Works, LLC
SVolume (3) SAttrDef (4) S8tmap (6) S8oot (7) S8adClus (8) S8adClus (8)	INDX entries for dire Total number of entri Total number of uniqu Only displaying 'slac Duplicates entries an Records displayed bas	<pre>ctory: [root]\\$Recycle. es found (w/ dups) w/o e (w/o dups) entries fo k' entries. e shown. ed on options selected:</pre>	Bin\S-1-5 any filte und w/o a	-21-1684986164-3766221403-; rs: 79 ny filters: 23
Secure (9)     Secure SSDS (9)     Secure SSDS (9)     SUpCase (10)     SExtend (11)     Secure SSDS (9)     Secure SSDS	Name: HFT Entry: HFT Sequence: Parent Entry: Parent Sequence: Entry: Modify time: Access time: HFTChanged time: Create time: Size allocated: Size used: Type entry: Source type:	<pre>\$IXC6CDV.tests ccorrupted&gt; ccorrupted&gt; ex000000003d2c [15660] ex0002 [2] slack, deleted 04/14/2014 03:02:53.19 04/14/2014 03:02:53.19 04/14/2014 03:02:53.19 04/14/2014 03:02:53.19 ex00000220 [544] ex00000220 [544] ex00000220 [544] ex00000220 [544] ex00000220 [ARCHIVE] indx alloc</pre>	2 [UTC] 2 [UTC] 2 [UTC] 2 [UTC]	

# 7 Available data for selected MFT record

**gena** is very useful for displaying file data. Not only does it show the normal data (unnamed data stream) and the alternate (named) data stream, but it also shows the Bitmap data, Logged Stream data, and various types of INDX data. A good example is looking at the \$Secure inode (#9). It has a number of attributes that contain data. Without going into too much detail, it contains: (a) an \$SDS alternate data stream, (b) \$SII index root and allocation attributes, (c) \$SII bitmap attribute, (d) \$SDH index root and allocation attribute. While some of the attributes might only have a little bit of data, or the data may reside as resident data within the file record itself, **gena** will display a separate tab for each data category. See below:

SBadClus (6)     SBadClus SBad (8)     SStoure (9)     SStoure (9)     SUpCase (10)     SUpCase (11)	Size stata Alloc size: 0x0 Valid size: 0x0 Slack size: 0x0	0040000 0040000 0000000	Data option Input: Goto Find	i (offset/search/ex 0x0000000 String FindNext	port) 30 Export	Show data as F Hex C Asci C WideAsci C UTF-16	Start • End
SQuota (24) SObjd (25) SReparse (26) SRmilletadata (2 SUsnJml (9413 SUsnJml5J) (94 SUsnJml5Max SRecycle Bin (3605 S-1-5-21-17302	0000 0000: 49 0000 0010: 00 0000 0020: 05 0000 0030: 00 0000 0030: 00 0000 0050: 50 0000 0050: 50 0000 0050: 50 0000 0050: 90 0000 0050: 90 ntfswak metadat	4c 44 55 e0 00 00 ef 00 00 e0 49 00 00 14 00 25 00 00 51 00 00 00 14 00 c3 01 00 95 00 00	21 00 09 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	ac 36 03 ad 0 28 00 00 00 3 ee 00 00 00 0 30 00 00 00 0 30 00 08 00 0 8c 2b 00 00 4 30 00 08 00 0 8c 23 01 00 4 30 00 08 00 0 8c 23 01 00 4 30 00 08 00 0 8c 23 01 00 5 bc 00 00 00 4	00 00 00     01 00 00     01 00 00     00 00 00     00 00 00     00 00	INDX(6 I I * * p	

Within each of these data tabs are options to go to a specific offset, find a string or pattern, or export the data to a separate file to analyze with another forensics tool. In some cases, it may be useful to represent the data as ASCII or Unicode instead of hexadecimal, so those options are there as well.

## 8 Utilities Menu

The Utilities menu was meant to be a catch-all location for helpful utilities. In this case, there was only one we had time to add to this and it includes the option to scan the mounted drives on a live system.

<mark>gn</mark> ge	na - (Graphical	Engine for NTFS Analysis)
<u>F</u> ile	<u>D</u> ata Format	Utils
	1511-	Scan Mounted Drives

When run, it will enumerate all the drives on a system and return a dialog box showing the drive, drive#, drive size, disk signature, partition information (offset, size and type), etc. This is helpful when using *gena* to analyze a drive, or image of a drive with multiple partitions, and one wishes to find the volume offset.

drive_num	drive_type	media_type	disk_guid	disk_sig	type	start_offset
9	disk	12	C4180403-0300-4697-0090-959179719035	7750-8257	GPT	exe
e	disk	12	C4180403-03ee-4697-aa9a-959179719e35	7750-8257	GPT	ex18888
8	GISK	12	C4f80403-03ee-4697-aa9a-959t79719e35	7750-8257	GPT	exe658888
9	disk	12	C4t80403-03ee-4697-8898-959t79719e35	7750-8257	GPT	exe75eeee
9	disk	12	C4180403-03ee-4697-8898-959179719e35	7750-8257	GPT	exeac+20000
6	disk	12	C4f8d4d3-d3ee-4697-8898-959f79719e35	7750-8257	GPT	ex1d15e8eeee
9	disk	12	c4t8d4d3-d3ee-4697-aa9a-959t79719e35	7750-8257	GPT	8x1d15898888
e	disk	12	c4f8d4d3-d3ee-4697-aa9a-959f79719e35	7750-8257	GPT	ex1d1c878888
		a - 20044				
1	015K	11	96599669-9666-9966-6966-666666666666	769-0138	MBR	8x8186666
			and the set of the second discon-			
2	disk	12	6673C82e-2038-420C-9095-97897800598C	078C-Da05	GPT	exe
2	disk	12	6673c82e-2b30-42bc-9b95-978978dd598c	078c-bab5	GPT	0x440
2	disk	12	6673c82e-2b3e-42bc-9b95-978978dd598c	078c-bab5	GPT	exe888446
2	disk	12	6673c82e-2b3e-42bc-9b95-978978dd598c	e78c-bab5	GPT	exes10000
2	disk	12	6673c82e-2b38-42bc-9b95-978978dd598c	078c-bab5	GPT	0xf42c10000
2	disk	12	6673c82e-2b30-42bc-9b95-978978dd598c	078c-bab5	GPT	0x1dcee90000
Wount poin Volume: RUID: Symbolic L	C: \\?\Vol ink: \\.\Har	une[d66c534f-i ddiskVolune3	dcbb-4f6d-a4a8-6a1a8245c698}\			
inlume:	D.					
SUID: Symbolic L	<pre>\\?\Vol ink: \\.\Har</pre>	ume(67d66c8b- ddiskVolume6	f17e-11e8-b983-185680762356}\			
				Ĩ		
/olume: SUID:	E: \\?\vol	ume{728c0d83-	9855-4d5d-92d7-9438f8b4a6c6)\	7		
(	Embra 15 Stringe	dal Carlottin Trans. P				

## 9 Use-Cases

There are a couple of *use-cases* that **gena** and **ntfswalk** provide. Below are some of the more common ones.

## 9.1 General Analysis on Targeted files

The first is general analysis on a specific file. The ability to pull all the NTFS attributes and present them to the user for analysis or exportation is easily available with just the *gena* application. Much of the discussion in this users guide goes over the different options available to the user. This is great for reversing a new file type or structure within the NTFS file system.

## 9.2 Incident Response Collection

The second use, which is more applicable to incident response, handles the case of extracting many types or category of files while the target machine is still running into a separately mounted hard drive. Where some client machines need to stay on to minimize impact to operations, this approach allows only the necessary files to be extracted. *gena* offers two live collection options: (a) to run *ntfswalk* separately from a script generated by *gena* beforehand or (b) to run *gena* directly which can spawn

*ntfswalk* to perform the collection. In either case, both tools can be run from a USB drive and the collection results can be sent to a network share or separately mounted drive.

As the collection has been completed on the desired target machines, one would perform analysis offline on the captured results.

## 9.3 Generating Hash Sets on Targeted File types

There are a number of excellent tools present on the Internet that perform hashing and creating hash sets. While *ntfswalk* was not designed to generate hash sets, it does have the ability to hash any desired target file. The main difference between *ntfswalk's* approach to that of a normal hash tool, is *ntfswalk* accesses the file contents at the cluster level whereas many other hashing tools do not. This becomes more important when considering our age of malware, and whether the actual file contents we are viewing have been masked by malicious software.

In using **gena** as the front end, one can, for example, select all the files with a signature used in executables, device drivers or libraries, and select the menu option "Hashfile" under the "Final output".



After spawning *ntfswalk*, the results file will contain a list of executable files with their computed hashes. Both the MD5 and SHA1 hashes are computed per file processed. The file generated will be named *ntfswalk\_hash\_results.txt* and use a pipe character ('|') as the field delimiter.

Depending on interest, adding more flexibility to *ntfswalk* on hashing options can be added in the future.

## 9.4 Support of Collection into Timeline Analysis

*ntfswalk* includes options to put the results file into a number of formats that can be easily manipulated programmatically where their output can be put into many timeline analysis packages.

A popular choice is to use the *Log2Timeline* tool. *ntfswalk* uses the older *Log2Timeline* format version 2 when rendering the data. If desiring to use the *sleuthkit*, one can use the *Body-file* format as an option. If using another package, the default or CSV options have the most fields that are easily parsable.

## **10 X-Window Dependencies**

For this tool to work, the X Window System libraries are required for both Linux and macOS (they are not required for Windows). These libraries use the X11 protocol and graphics primitives to render the graphical user interface components. These libraries are common on Unix-like OS's.

If one is unfamiliar with X Windows or the libraries associated with it, one can download an installer package from XQuartz.org, which is an open-source effort to develop a version of the X Windows System that runs on Linux and macOS.

After the X11 libraries are installed, one needs to ensure they are running prior to running this tool.

## **11** Authentication and the License File

This tool has authentication built into the binary. The primary authentication mechanism is the digital X509 code signing certificate embedded into the binary (Windows and macOS).

The other mechanism is the runtime authentication, which applies to all the versions of the tools (Windows, Linux and macOS). The runtime authentication ensures that the tool has a valid license. The license needs to be in the same directory of the tool for it to authenticate. Furthermore, any modification to the license, either to its name or contents, will invalidate the license.

## 11.1 Limited versus Demo versus Full in the tool's Output Banner

The tools from *TZWorks* will output header information about the tool's version and whether it is running in *limited, demo* or *full* mode. This is directly related to what version of a license the tool

authenticates with. The *limited* and *demo* keywords indicates some functionality of the tool is not available, and the *full* keyword indicates all the functionality is available. The lacking functionality in the *limited* or *demo* versions may mean one or all of the following: (a) certain options may not be available, (b) certain data may not be outputted in the parsed results, and (c) the license has a finite lifetime before expiring.

## **12 References**

- 1. FOX-toolkit version 1.6.47, GUI
- 2. TZWorks, LLC, *ntfswalk*, <u>http://www.tzworks.com/prototype\_page.php?proto\_id=12</u>
- 3. TZWorks, LLC, *wisp*, <u>http://www.tzworks.com/prototype\_page.php?proto\_id=21</u>
- 4. TZWorks, LLC, *ntfscopy*, <u>http://www.tzworks.com/prototype\_page.php?proto\_id=9</u>
- 5. TZWorks, LLC, *ntfsdir*, <u>http://www.tzworks.com/prototype\_page.php?proto\_id=8</u>
- 6. <u>http://en.wikipedia.org/wiki/NTFS</u> website
- 7. Brian Carrier's book, File System Forensic Analysis, sections on NTFS
- 8. Various Microsoft Technet articles.
- 9. <u>X Window System Libraries</u> by XQuartz.org.