TZWorks® 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.

www.tzworks.com

Contact Info: info@tzworks.com

Document applies to v0.60 of ***gena***

Updated: Apr 25, 2025

Table of Contents

TZWorks® NTFS Analyzer/Viewer (*gena*) Users Guide

Webpage: http://www.tzworks.com/prototype_page.php?proto_id=28
Contact Information: info@tzworks.com

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*

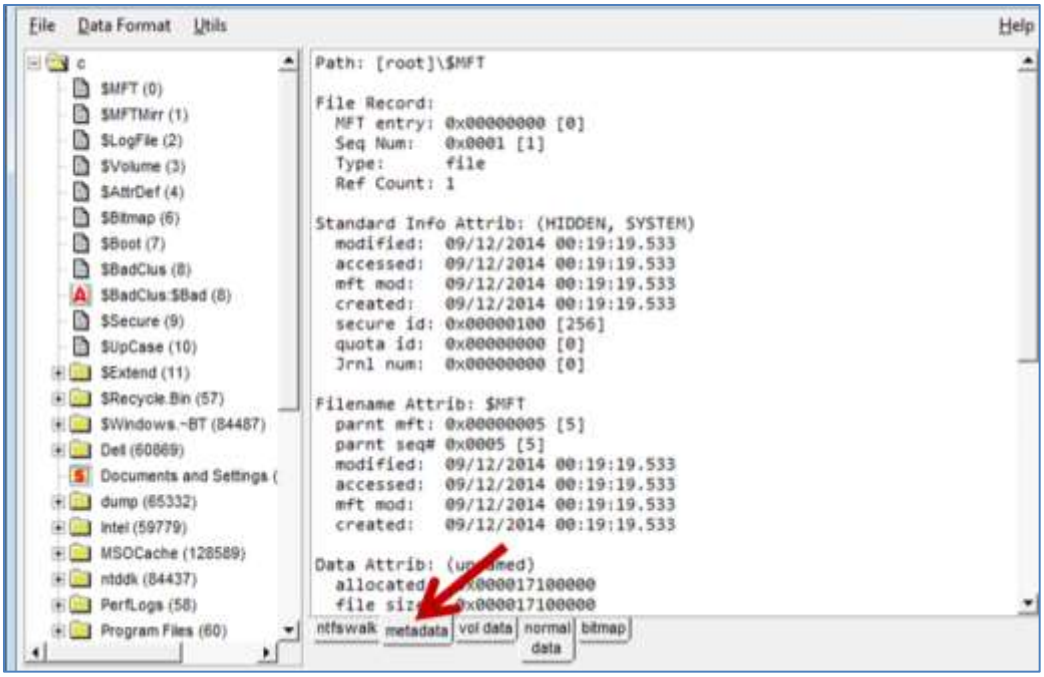
The ***gena*** GUI is divided into two main window panes. The first pane is on the left, and is a quasi-Windows Explorer view of the volume under analysis. The right pane is governed by a series of tabs that reflect differing functionality. The number of tabs is dependent on the MFT record (or *inode*) 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.

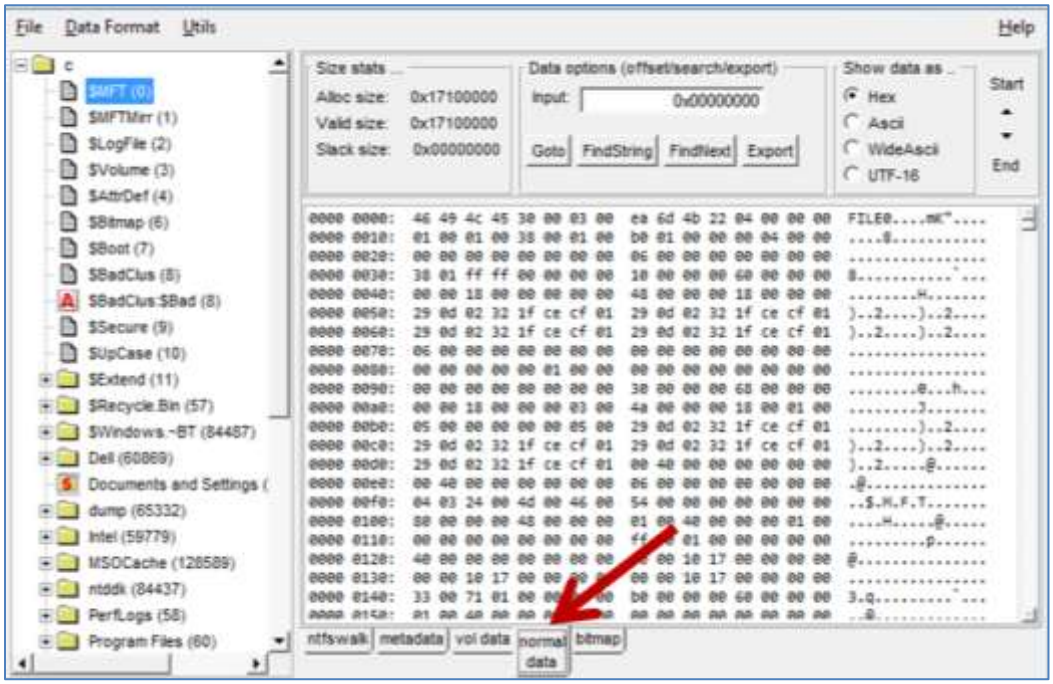
Below is a screenshot of analyzing a 'C' partition from a local machine while the *$MFT* file is selected. When looking at the directory tree, one can see any alternate data streams via the superimposed red letter 'A' on the file icon. This is only an artifact of ***gena*** to flag to the user that an 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.



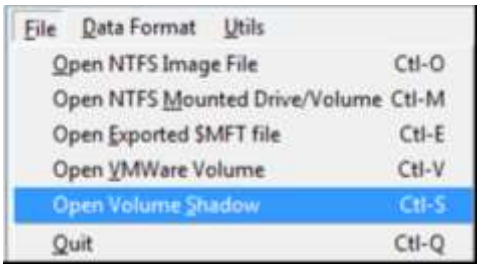
To look at the cluster runs for any data attribute, one need only select the appropriate data tab. The data tabs, have a number of options available, such as: (a) jumping to a specific offset in the cluster run, (b) finding a string or pattern, (c) exporting the data section to a separate file, (d) showing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



2 Source, Results File, Data/Time Options

2.1 Source Options

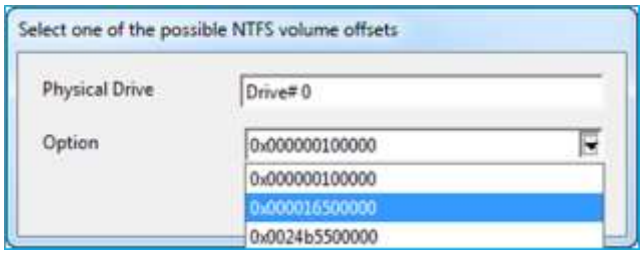
Currently, there are five types of source data ***gena*** can analyze, including: (a) NTFS drive/volume stored as a 'dd' type image, (b) NTFS mounted drive/volume, (c) an exported $MFT file, (d) a VMWare volume, or (e) a Volume Shadow Copy. To analyze an image or mounted volume, one uses the "File" menu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



2.1.1 Examining a Mounted Volume or Drive Number

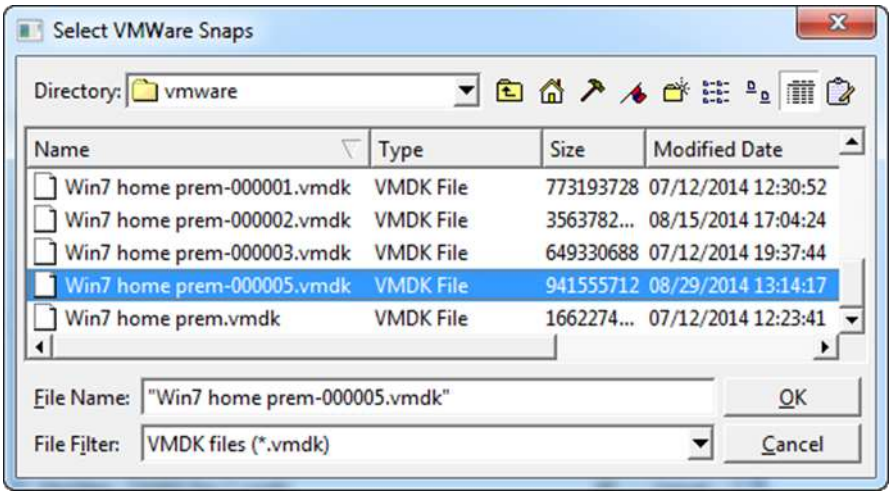
By selecting the second option in the File Open options, one can select either a mounted volume letter or a mounted drive number. The following dialog is displayed after the initial menu selection:

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.

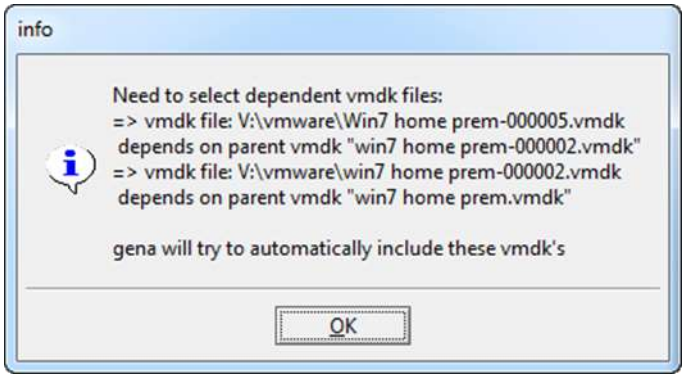


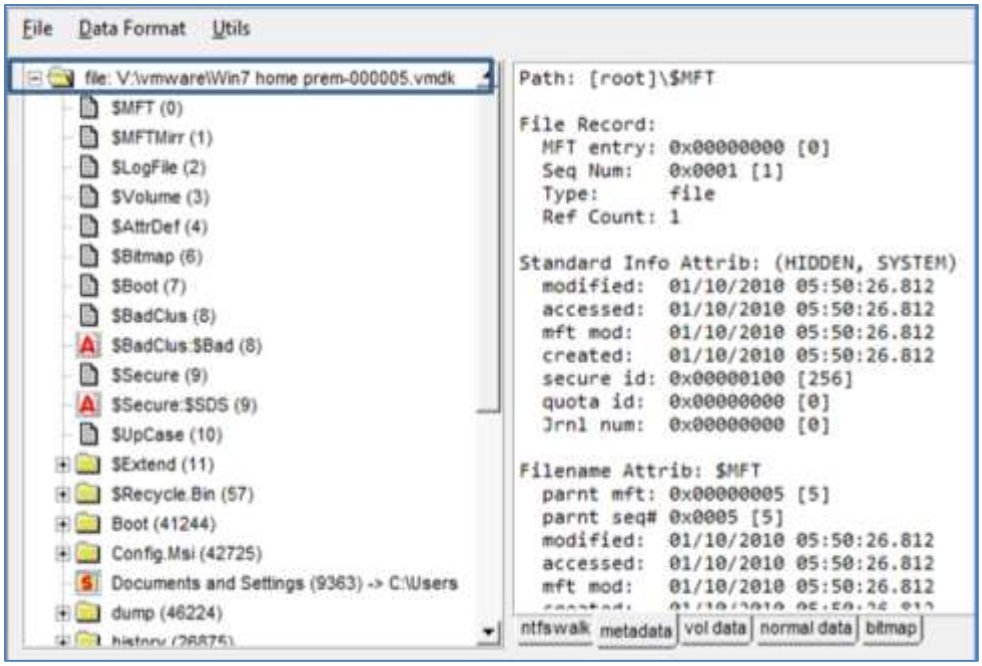
2.1.2 Examining a VMWare Volume

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 5th snapshot as shown below, causes ***gena*** to look at any dependencies available.



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.



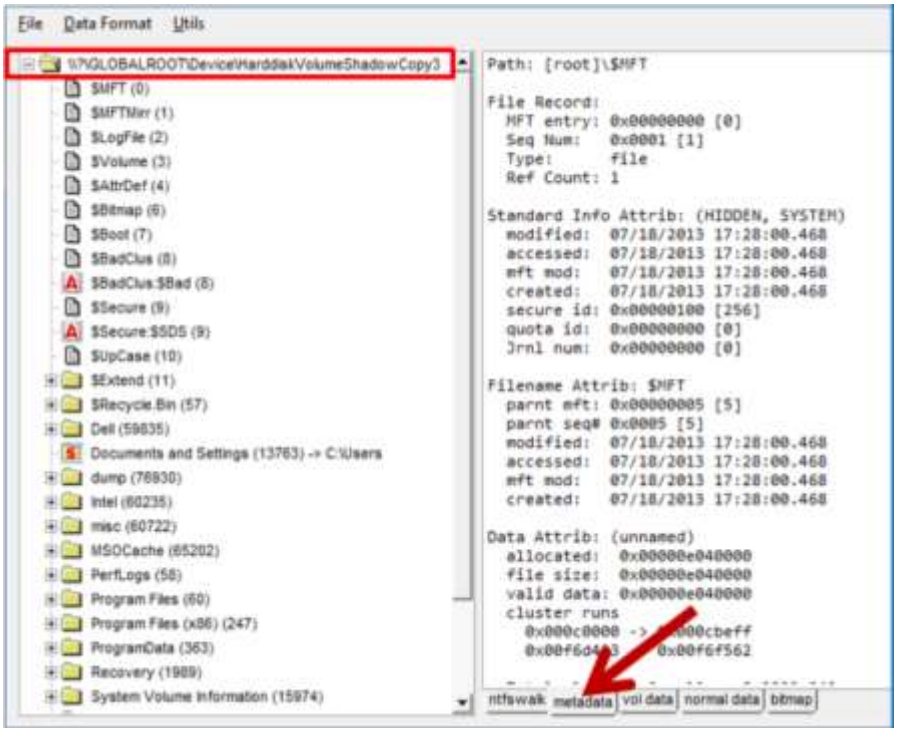


2.1.3 Examining a Volume Shadow

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

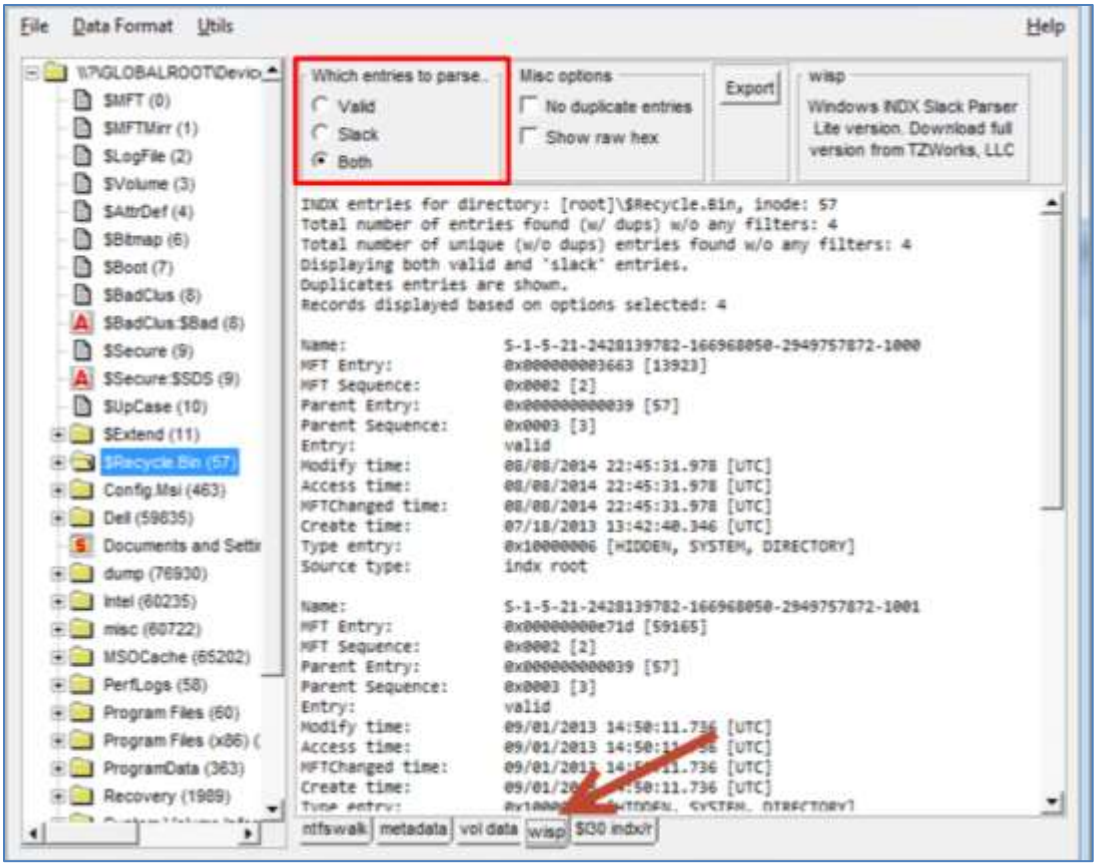
When selecting Volume Shadow as the source, one will be asked which Volume Shadow Copy to analyze. Each Volume Shadow has an index. ***gena*** will automatically enumerate all the available indexes and display then in the dialog box below. This index equates to the *HarddiskvolumeShadowCopy#,* where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

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.



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.

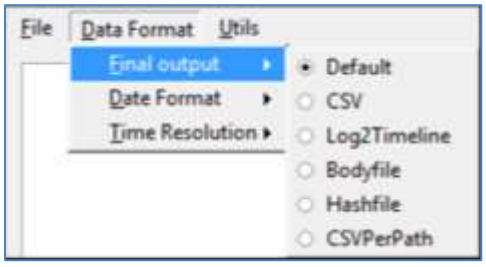


2.2 Format options

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 http://log2timeline.net/. 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.

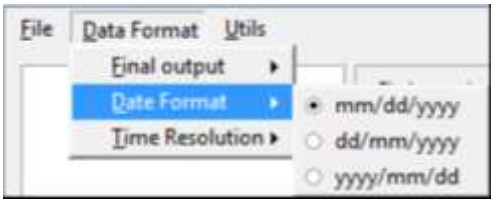
The *Hashfile* format will generate both an MD5 and an SHA1 entry as well as other metadata per *inode* that was processed.

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.

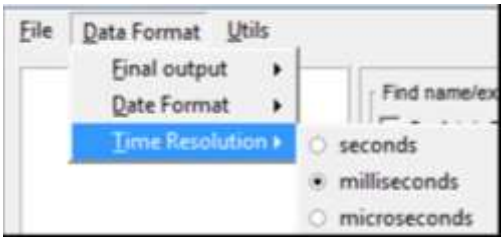
2.2.2 Date format

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:



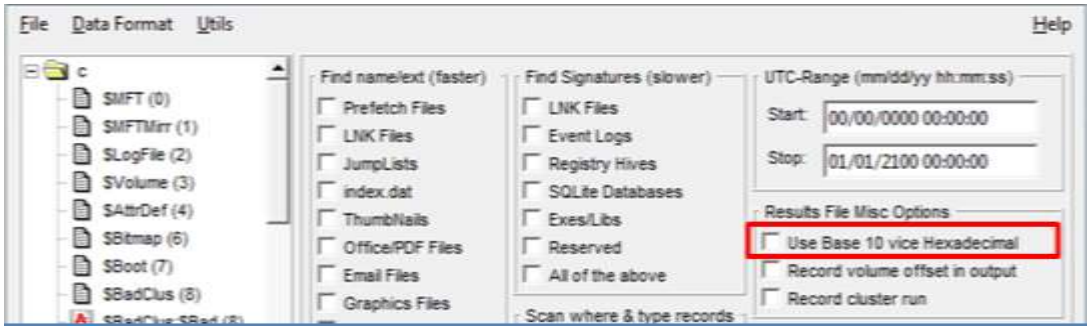
2.2.3 Time format

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.



3 *ntfswalk* dialog tab

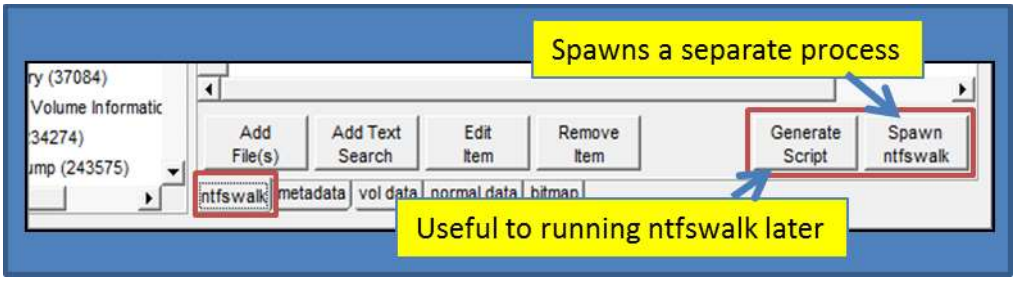
gena was originally designed to be a front end to the ***ntfswalk*** command line tool. The additional capabilities added to version 0.45 of ***ntfswalk*** coupled with **gena** make it easy to target specific files in a scripted manner, whether for incident response or normal forensic data collection. ***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

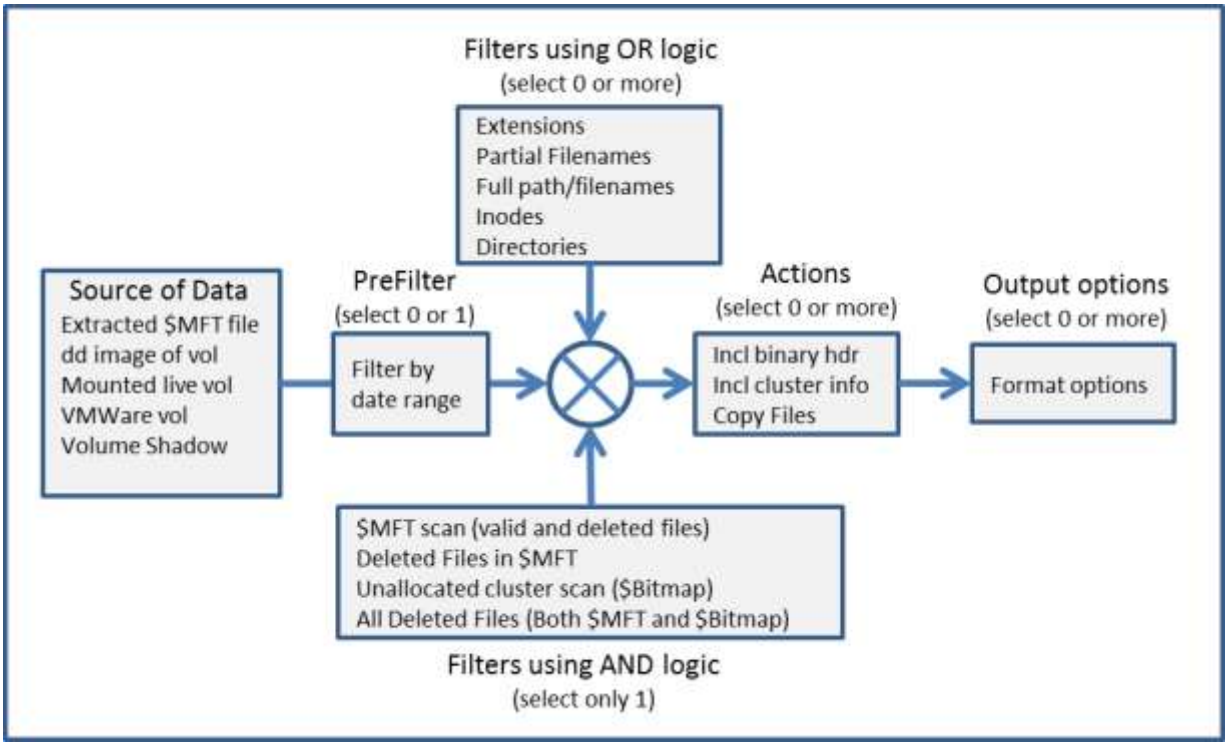
process passing to it the options the user has selected. ***gena*** and ***ntfswalk*** really do not care whether the target volume is a live mounted image or a 'dd' image. They process them the same way. As an aside, since ***gena*** will spawn ***ntfswalk***, both tools should be in the same directory 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.

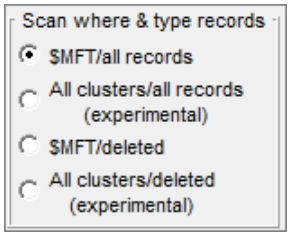
All external filtering options are shown by the checkboxes grouped via the “Find name/ext (faster)” title and shown below. One can select none, one or more options to filter on.

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.

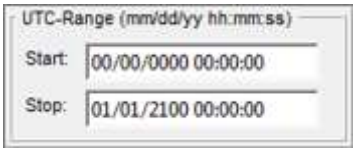
3.1.3 Selecting what clusters to scan and whether we are just interested in deleted records

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).



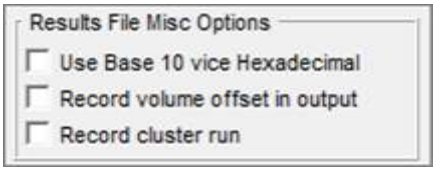
3.1.4 Selecting Files in a Time Range

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.

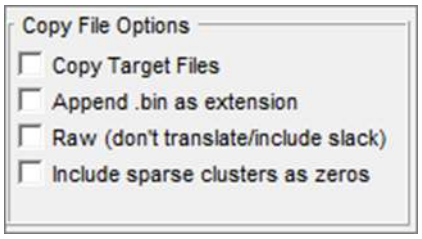


3.1.6 Extraction Options

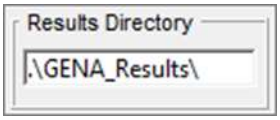
If desiring to extract files to an archive directory, one can select the “Copy Target Files” checkbox. If one is worried about copying malicious data from a target box one can change the extension of the file extracted by appending a *.bin to the file, so it is not accidently run should the file contain 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’

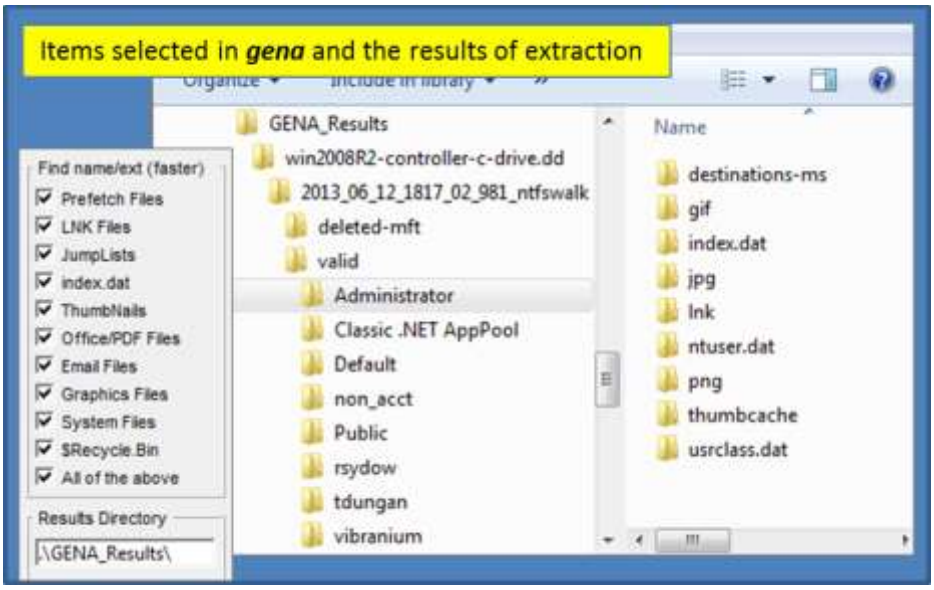
clusters. Therefore the 'raw' option will yield the slack that is available. The last option is whether one wishes to include any sparse clusters as part of the file copy. The sparse clusters, while present on some files, are not backed by physical clusters and therefore ***ntfswalk*** will copy zeros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



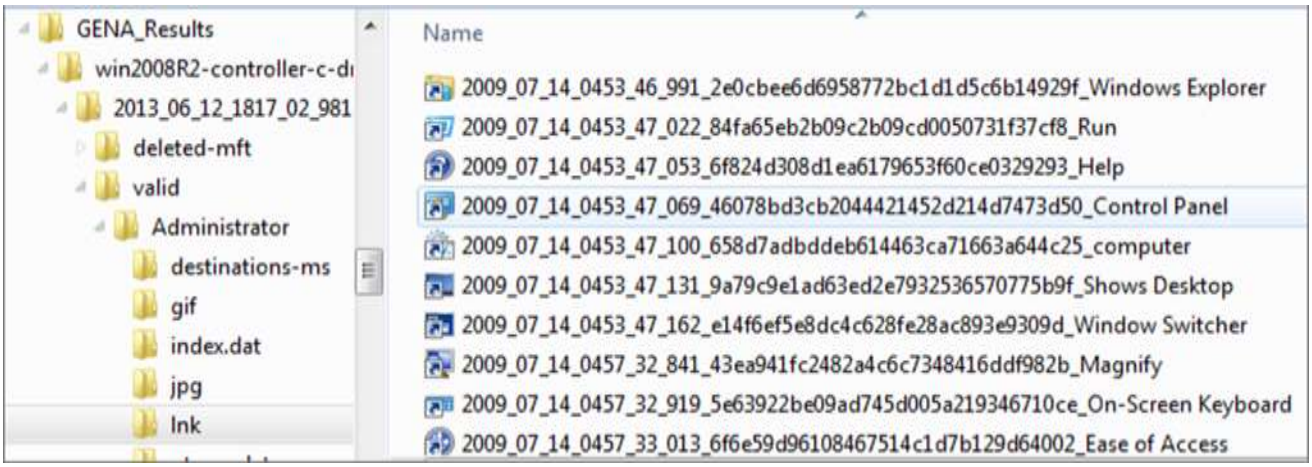
The extraction directory for both the results file and any files extracted is specified by the text entry box below. The default directory is "GENA_Results" under the current working directory. Whatever directory is inputted here will be used as the parent directory for archiving data. ***ntfswalk***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



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.



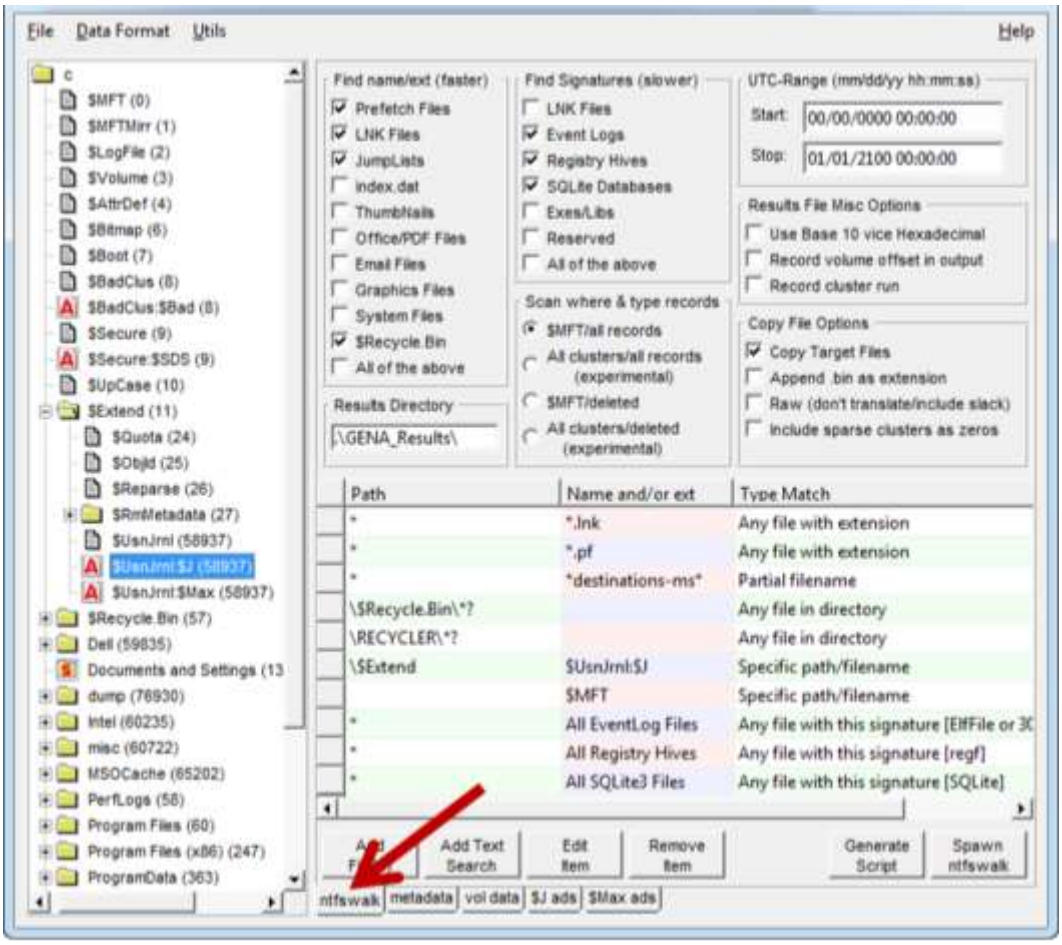
3.1.7 Manually adding and deleting filters

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.

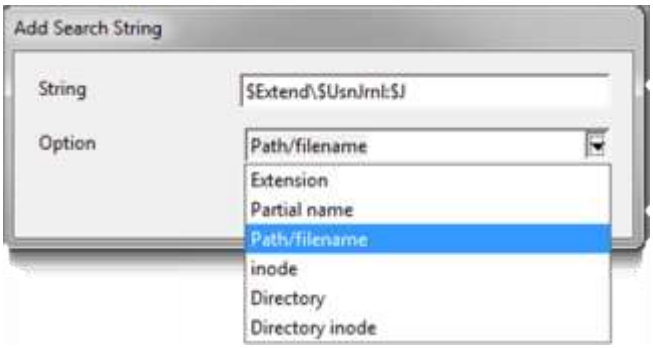
As an example, look at the snapshot below. In the example, *Prefetch files, LNK files, Jump Lists* and *$Recycle.Bin* are selected from the first set of filters. Looking at the list box, these show up as **.pf, *.lnk, *destinations-ms** and *$Recycle.Bin*?* entries, respectively. The first two are filters 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.



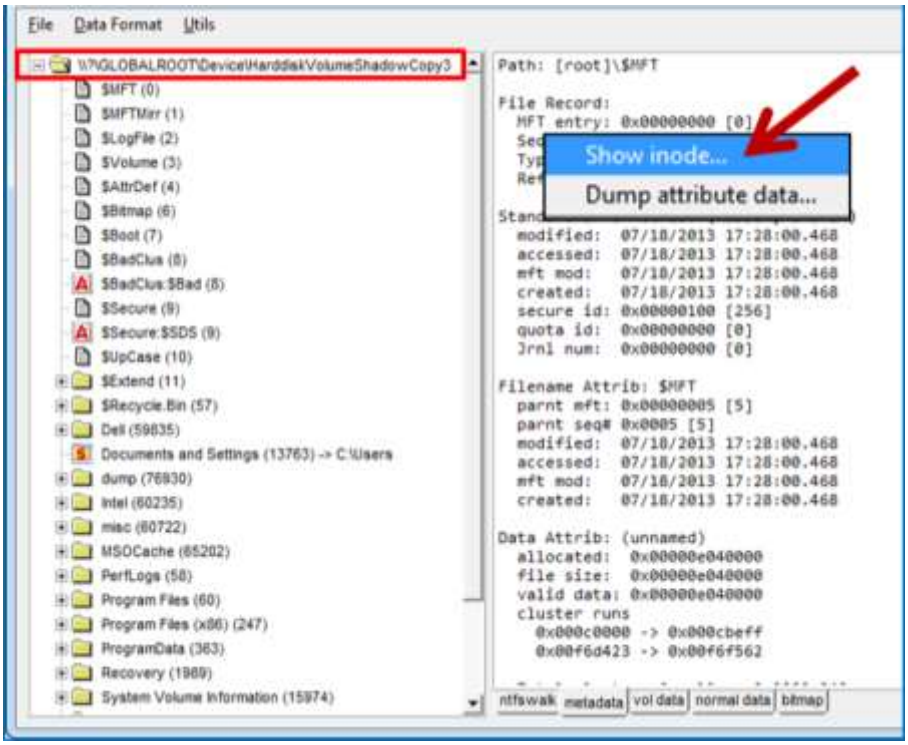
The first option is the “Add File(s)” button. This is used when selecting (highlighting) one or more items from the directory tree on the left pane and pressing this button. The selected entries will be added to the list box. The second option is the “Add Text Search” button. This offers the user some 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.



4 MFT record metadata tab

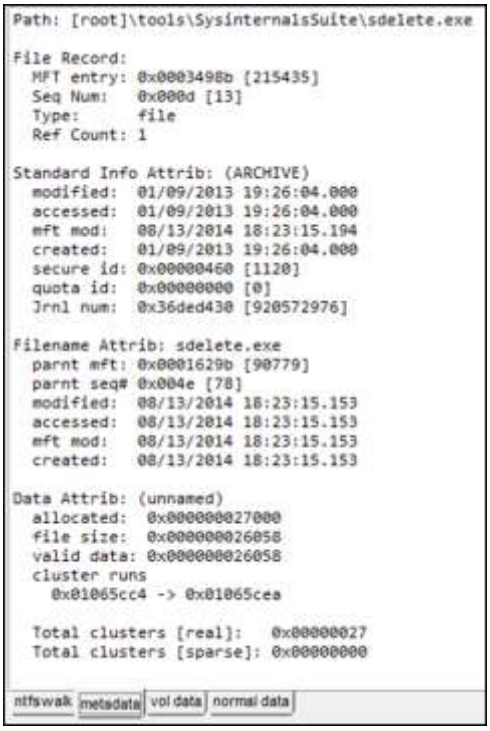
The metadata in this tab includes an 'interpreted' version of the NTFS file record. This means that, if ***gena*** sees a standard information attribute, it will show the properties and timestamps associated with this sort of NTFS attribute. Likewise, if ***gena*** sees a filename attribute, it shows 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.



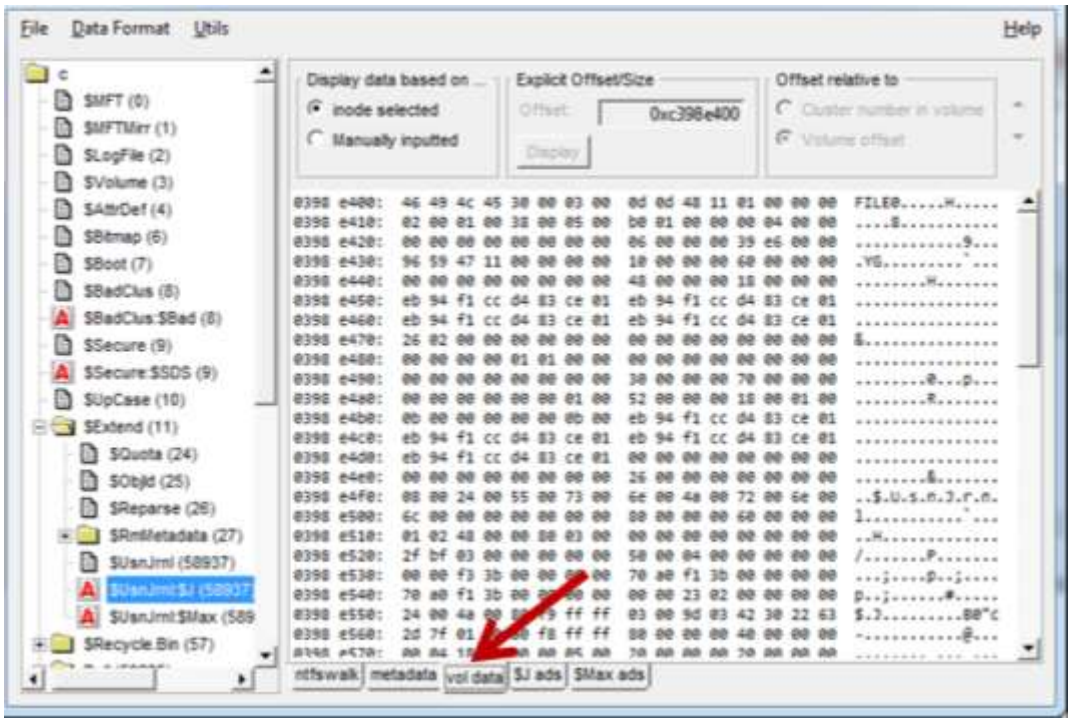
By selecting "Show inode", a dialog box will pop up allowing one to enter the inode value they wish to retrieve. For example, let's say one wanted to evaluate what file was associated with inode 215435, one could enter this value (or the hex value 0x3498B) into the dialog box and it would retrieve the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

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.



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.

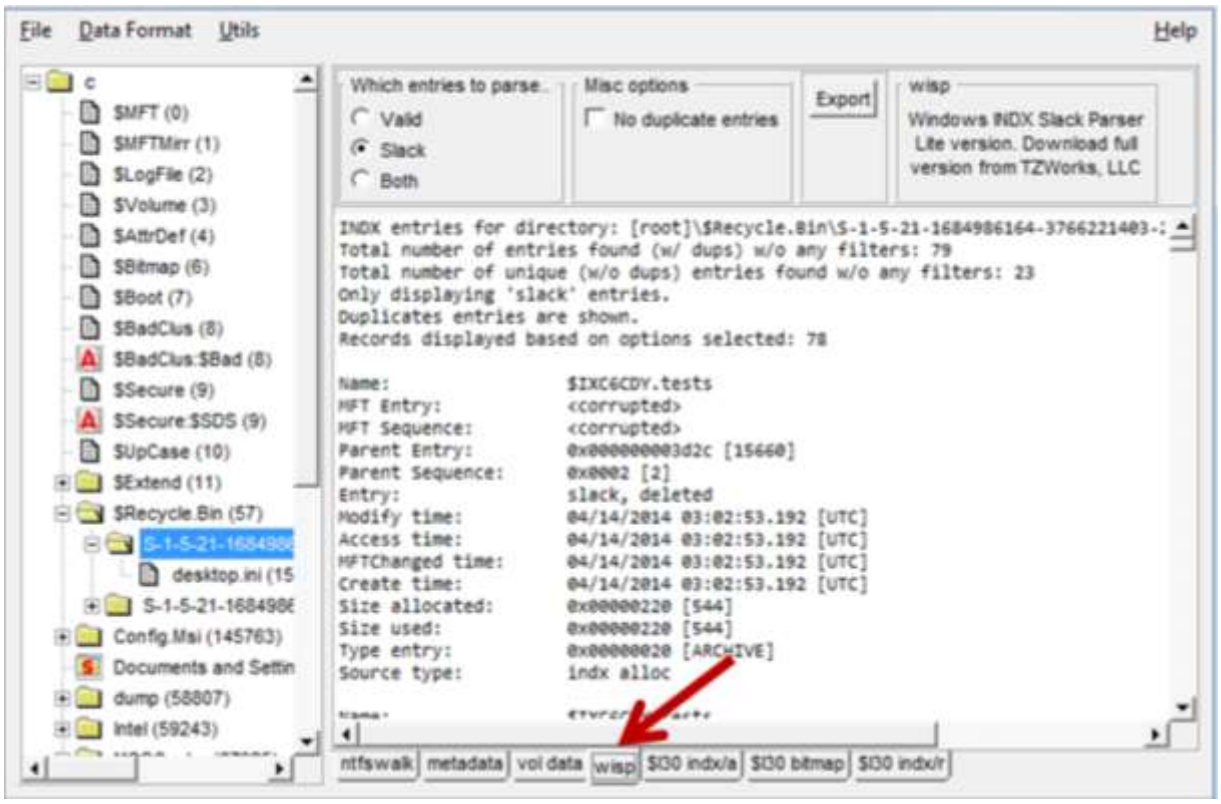


With this tab, one has the freedom to look at any volume offset that is desired. To do this, just select the "Manually inputted" radio button and the offset field will be activated as well as the display button. One has the option of traversing the volume either by volume offset address or by volume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

6 *wisp* tab

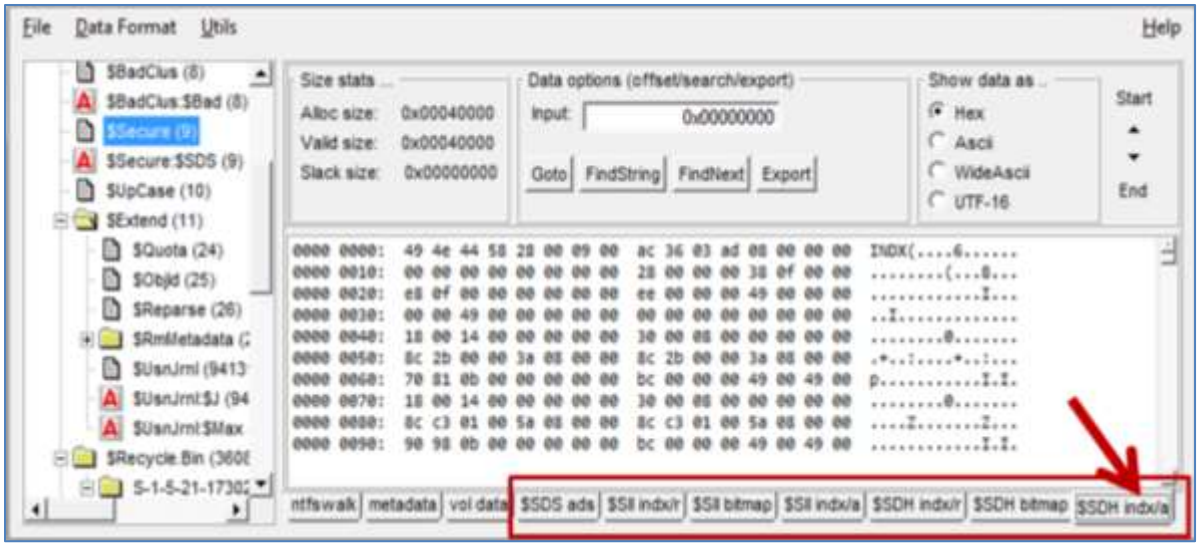
When selecting any directory from the tree view, a ***wisp*** (Windows INDX Slack Parser) tab is displayed. Since ***gena*** can parse the INDX attribute to find the children to a directory, we incorporated a lite version of the TZWorks ***wisp*** engine. This will allow ***gena*** not only to parse 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.



7 Available data for selected MFT record

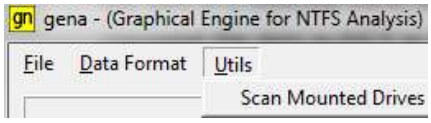
gena is very useful for displaying file data. Not only does it show the normal data (unnamed data stream) and the alternate (named) data stream, but it also shows the Bitmap data, Logged Stream data, and various types of INDX data. A good example is looking at the $Secure inode (#9). It has a number 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 attributes, and (e) an $SDH bitmap 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:



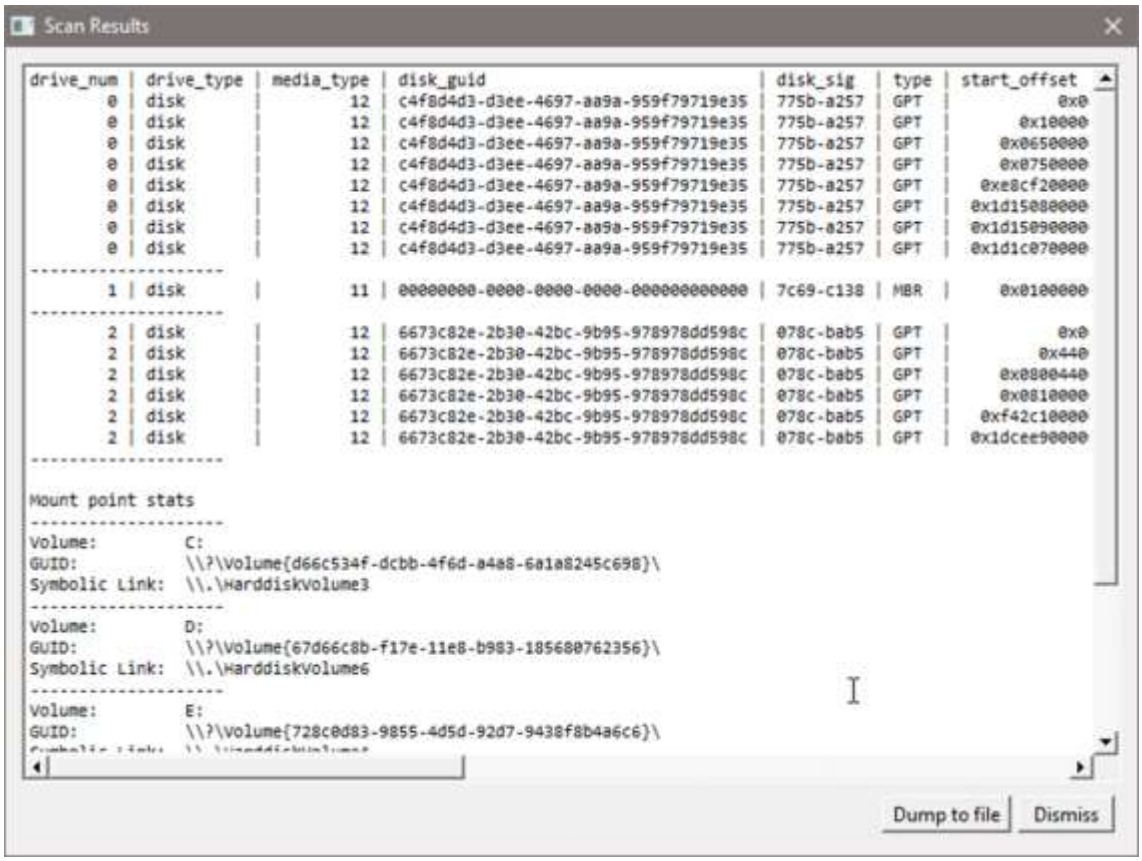
Within each of these data tabs are options to go to a specific offset, find a string or pattern, or export the data to a separate file to analyze with another forensics tool. In some cases, it may be useful to represent the data as ASCII or Unicode instead of hexadecimal, so those options are there as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



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.



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

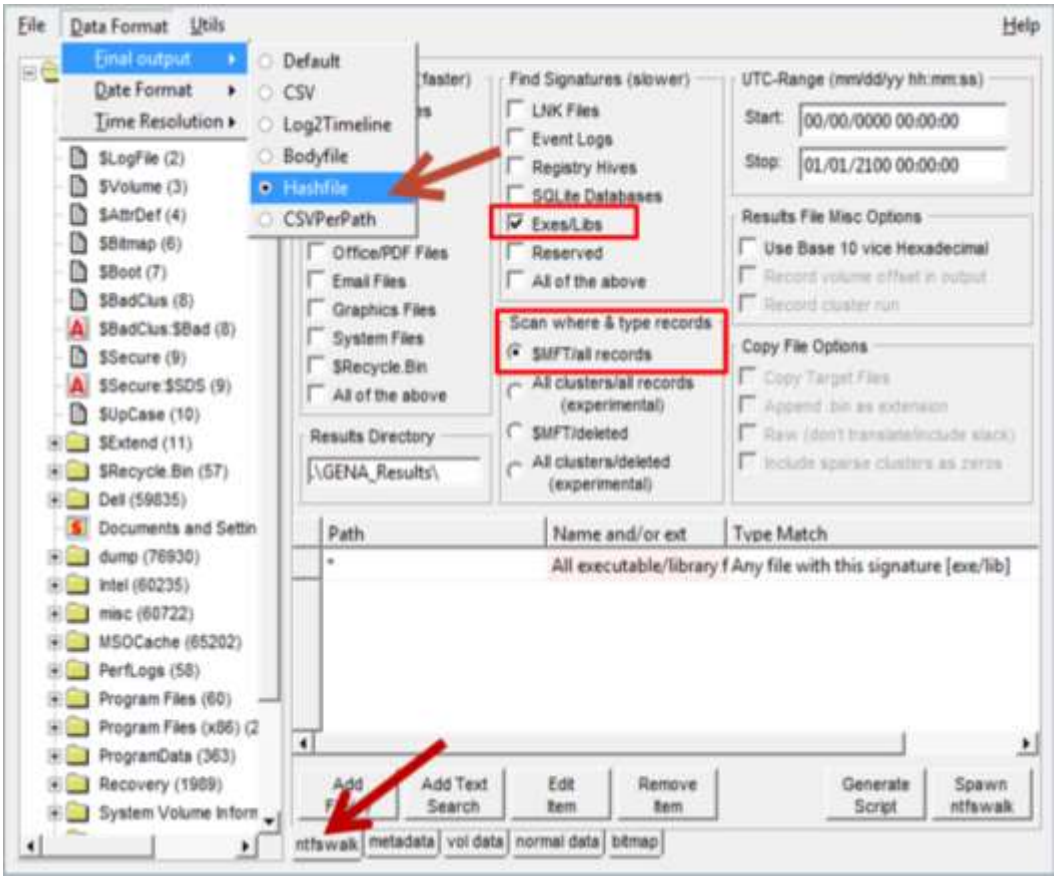
ntfswalk to perform the collection. In either case, both tools can be run from a USB drive and the collection results can be sent to a network share or separately mounted drive.

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***, http://www.tzworks.com/prototype_page.php?proto_id=12
3. TZWorks, LLC, ***wisp***, http://www.tzworks.com/prototype_page.php?proto_id=21
4. TZWorks, LLC, ***ntfscopy***, http://www.tzworks.com/prototype_page.php?proto_id=9
5. TZWorks, LLC, ***ntfsdir***, http://www.tzworks.com/prototype_page.php?proto_id=8
6. http://en.wikipedia.org/wiki/NTFS website
7. Brian Carrier's book, File System Forensic Analysis, sections on NTFS
8. Various Microsoft Technet articles.
9. X Window System Libraries by XQuartz.org.