This hands-on practical session is designed to determine if:

This is a +- four hour hands on (or self-paced) practical session which will enable you (the analyst) to verify for yourself that the software you are using to **catalog, hash, copy, and zip** evidence files does or **does not** follow good evidentiary processes.

**1. You are requested to perform the analysis on a Windows10 or 7 computer with NTFS file system.**

**2. Prior to the class, you will be provided the links to a few executables to download. You will put these executables on an 8G NTFS formatted thumb drive, or in a separate folder on your work computer. These executables contain: original evidence, various sample software to use, and data to use to test the capability of your software.**

**3. On the day(s) of the class, you will be provided the password and command to restore the appropriate test data and software to its working state. All the executables you are provided are encrypted with different passwords which will be distributed when needed.**

You will be advised which directories extracted to the drive should be treated as original evidence. Other directories are provided which contain sample software to install and test, and other directories which contain various administrative items.

Analysts are encouraged to bring their own cataloging, hashing, copying software and are encouraged to perform certain tests using their own software first, then use the software provided (over 40 different programs) to perform cataloging, hashing, copying, and zipping of evidence provided on the media. And to record the results for others to see.

=================================================

All the information/suggestions relating to the setup and execution of the batch files assume (you know what that means) that the testing and data resided from the root of the thumb drive. Operation from any other directory will require the user to modify the paths associated in the batch files to point to the correct locations.

Once the files are extracted from the executables, open the \_RESET\_PATH.bat files and adjust the path to reflect the path which the **SOFTWARE\MWARE** is set to. This will reset the system path so that the Maresware software, will be in the appropriate path and will run as designed.

Again, after the **\_RESET\_PATH.bat** has been properly adjusted, run it to reset the path which will allow you to run the necessary Maresware software to confirm or deny that your runs corrupted the evidence or not. If you don’t reset the path, then none of the batch files setup to confirm file integrity will work.

Then, at some point, go to the session which instructs how to reset the last access date update in the registry. This is a very important step which will impact most of the testing requirements.

=================================================

The sessions are set up into four categories. The sessions may be completed on three or four separate days, or in one day in about 4 hours. The order of operation is described below. It is necessary to perform the tests in the prescribed order so as to allow you to properly understand the testing processes and that of confirming file counts in evidence.

Also, if you use any cataloging, hashing, copy, or zipping software in your day to day activities, you should at a minimum perform some of the tests using **your own familiar software** before using other software that is provided or downloaded.

The tests will ask you, the analyst to confirm that the software you have used and tested has properly processed the data provided, and did it in a forensically sound manner, without altering any content or meta data of the evidence.

=================================================

Proper forensic practices relating to the test topics will be discussed at the beginning of the session the first day (hour). The following topics will be discussed as requirements for the tests.

* The purpose of maintaining original file dates on source and destination when
  + file listing/cataloging a tree structure.
  + hashing.
  + copying.
  + zip and unzipping evidence.
* Long filenames (>255) in the Windows environment, and need to properly maintain the path/filename.
* Alternate Data Streams in the Windows environment and need to properly handle them.
  + A routine action that may cause ADS’s to be created.

Bottom line

These tests will confirm or deny the programs ability during cataloging, hash, copy, zipping process to:

* Properly count and identify all the files within the evidence tree.
* Maintain source/destination file dates while performing the specific operation.
* Maintain source/destination long file names.
* Maintain source/destination alternate data streams.

Analysts will work at their own pace during the time available, and obtain assistance, suggestions and input from others in the room when necessary.

See the **TESTING\_PROCESS** document file for further explanation of the file structure on the disk.

=================================================

These are the four categories which will you will be asked to perform and evaluate your software to complete the evidentiary task. The stated tasks and requirements are loosely listed here.

1. File/tree/folder/directory (you get the idea) cataloging and listing files within the evidence tree structure. This test will be performed on the top level folder called **CATALOG\_EVIDENCE\FILES**. There is a simple test to complete in the ADMIN folder after some of the runs/tests are completed.

2. The forensic file hashing, copy and zipping software you use or have access to provide **true complete copies and processes/hashes of all the evidence files**. Suites are not recommended, although they can be tested if used to test against the tree structure. NO BIT IMAGES allowed in the test.

3. Confirm that your process or software **does NOT alter** in any way the original files content, location or Meta data. (dates, times, sizes, paths/filenames). Altering or **missing** evidence is considered as altering evidence.

4. The copy process (and zip/unzip process is considered a copy operation) can copy **all** the evidentiary data files to a destination while maintaining all original file structure and any meta-data without alteration.

There are four sections to this document.

**Background: (page 3)**

Background as to why I performed and designed these tests and requirements.

**First Step: Computer setup requirements: (page 4)**

Explains how you MUST set up your computer and test thumb drive for this session.

**Theory Behind These Tests: (Page 9)**

Forensic processes and theory about why you should consider testing your software.

**The Actual Processes: (page 5)**

The actual three part testing process.

Part 1: discuss the evidence process for the next 3 hours, and: perform file cataloging/listing,

Part 2: hash (day 2, hour2)

Part 3: copy (day 3, hour3)

Part 4: Zip/restore/retention. (day 3, hour3)

==================================

**Background:**

When cyber forensic investigations are conducted there are a number of processes and steps which may be considered as basic, standard, necessary and useful. These minimal steps will obviously vary depending on the type, scope, goal of the investigation and final disposition of the evidence found. However, much of this is open for discussion, agency/corporate standards and challenge by attorneys. So proceed at your own risk.

Whether you capture cell phone data, photo meta data, vehicle GPS, intrusion/network data, computer files in pornography, data exfiltration (corporate theft), or any other type of cyber investigation, I hope we all can agree at some point that data will end up on your forensic computer for additional analysis, reporting, and ultimate delivery to an adjudication authority, whether administrative or judicial.

Some “minimal” or basic processes conducted on the “original” or firstly acquired evidence might include the basic processes tested for here:

1. creating a full and complete catalog or list of all the files within the specified evidence tree/directory;
2. hashing for original state and data integrity;
3. forensic copying for data retention or original acquisition from the source drive/server; and
4. zipping (and unzipping) the evidence for future delivery to attorney, retention and/or retrieval by the adjudication party.

==========================

**Eventually You will devise your own set of tests** to determine if your programs and process can properly process the provided evidence files. However, for these sessions, you will be provided sample evidence files and folders, and the necessary requirements for your tests to complete.

Your tests will be conducted on the sample evidence/data provided and hopefully will show you that your own, and many “recommended” and routinely used forensic programs do not perform as expected and may leave themselves (the software) or your process open for evidentiary arguments.

Again, the tests (catalog, hash, copy, zip/unzip/restore) which you perform cover the areas which are likely and routinely used in computer forensics and evidence presentation and preservation.

**First Step: Computer setup requirements:**

**1. Before doing any of the tests, you must make sure that your WINDOWS computer has last access update turned on. Last access date maintenance is part of the testing process.**

To do this, make sure the registry keys are set as shown here: set it to 0 (zero). Ignore the other two options of setting to 2, or 3. They are not needed for this test.

**Key Name: HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem**

**Name: NtfsDisableLastAccessUpdate**

**Type: REG\_DWORD**

**Value: ~~1 (A value of 1 turns last access update off.)~~**

**Value: 0 (Sets last access update to on. Access dates are updated)**

2. Prepare a thumb drive of at least 8G in size with the NTFS file system, or have an empty NTFS folder ready on your work computer to receive the data.

NTFS is necessary for a majority of the testing to work. NO WRITE BLOCKERS ALLOWED:

**PEOPLE: READ MY LIPS!! or READ THIS:**

**A significant number of the tests require an NTFS file system. SO: Format the drive accordingly.**

If you wish, you can run these tests on a sub-directory of your choosing. But must be an NTFS file system.

*Running from a sub-directory will negate the operation of any of the test batches provided. You will have to alter their contents to make them work from sub-directories*.

3. Make sure you are running all processes as administrator.

4. You were (or will be) provided download links to the appropriate executables that will contain all the test evidence, documents, and sample software which you will use to run the tests. The executables will be encrypted, and you will be provided the passwords to extract the data at the appropriate hour or day.

The executables are (maybe different names depending on the version of this document):

**\_CATALOG\_EVIDENCE.exe**

**\_DEMO\_FILES.exe,**

**\_SOFTWARE\_CATALOG.exe**

**\_SOFTWARE\_HASH.exe**

**\_SOFTWARE\_COPY.exe**

They were created using the only zipping program I could find that would pass all my tests, and properly zip and unzip the required evidentiary/sample file structure. (*Finicky virus checkers will claim a virus. There is none*).

5. Place the encrypted files in the root of the thumb, or in the top of the chosen directory on your computer. The command below to extract the appropriate data is to be used. The appropriate password will be provided at the right time. Execute the following command (exactly) as provided.

**F:> \_xx\_PROGRAM.exe -s2 -ts+ -tsp -tp+ -os -ppassword\_will\_be\_provided**

It will extract the sample evidence and some other directories containing reference documents, and minimal software and batches which I have developed to help you determine if your process corrupts the evidence. The provided batch files will also help restoring your corrupted evidence. And it will get corrupted.

6. Before doing anything else, once the files are extracted, make sure you edit, then run the file

**F:> \_\_RESET\_PATH.bat**. (There are two leading underscore in the filename) It can be found in either the root, or the **RECOVERY\_TEST** folder. If you don’t reset the system path, then none of the batch files setup to confirm file integrity will work. But those are my tests, which you don’t have to use. You are encouraged to develop and test your own processes.

An alternative is to place all of provided Maresware executables in one of the other pathed directories on your system.

Take a look at the folders: ARTICLES, RECOVERY\_TEST, and TEST\_BATCHES. They contain some batch files, which when run will determine if your software correctly processed the evidence. Further explanation to come.

7. AGAIN: **Note on the extraction or download of the exe data. A few, non-main line virus checkers see the exe as a problem. Because they check for a small bit/binary sequence which they think is a virus signature. It is not. If you have a virus checker that says a virus. Use another. I’m not going to play games with over enthusiastic virus indicators.**

====================================================================

**What’s in the \_DEMO\_FILES, \_CATALOG\_EVIDENCE, \_SOFTWARE\_FILES**

The **RECOVERY\_TEST** directory contains batch files which will be used to “reset” the evidence folders to their original state once your process is determined to alter and fail any particular test. And it will. Before running any test of a particular software product, you will use these batch files to confirm the original evidence is pristine and properly set to conduct the next test. See the **PRIMARY\_TEST.exe** or **TEST\_xx.bat** files below.

**Bolded items are evidence folders/tree.**

04/12/2020 11:37 AM <DIR> ARTICLES

**01/06/2020 03:31 PM <DIR> D1**

**01/10/2020 10:30 AM <DIR> D2**

**01/18/2020 06:14 AM <DIR> D3**

**01/18/2020 06:14 AM <DIR> D4 (maybe also D5)**

**02/10/2022 01:53 PM <DIR> CATALOG\_EVIDENCE\FILES**

04/10/2020 01:53 PM <DIR> OUTPUTS

04/09/2020 08:43 AM <DIR> RECOVERY\_TEST

04/10/2020 07:15 AM <DIR> SOFTWARE

**12/30/2019 07:07 PM <DIR> SOURCE1**

**01/12/2020 05:04 PM <DIR> SOURCE2**

**01/12/2020 05:03 PM <DIR> SOURCE3**

**01/08/2020 03:43 PM <DIR> SOURCE4**

04/10/2020 12:08 PM <DIR> TEST\_BATCHES

04/09/2020 09:45 AM 817 PRIMARY\_TEST.BAT

04/09/2020 10:13 AM 1,957 ZCLEANUP.BAT

The **\_ \_README.\*** *(double underscore)* files in the root are for additional reference and are not necessary.

The **\_ \_RESET\_PATH.BAT** *(double underscore)* file should be adjusted and **run first**. It adds to the current path the  **SOFTWARE\MWARE** directory, so all the verification and other batch files have appropriate software available to help you validate your processes. (You can, if you wish, modify the system global path to include the directory. Using the this batch for these sessions, is probably better.)

All the batch files in the **TEST\_BATCHES** directory may need to be edited so that the Maresware software being called is properly found. Since I did not know what your paths will look like, you should review and edit all the batch files in the **TEST\_BATCHES** directory prior to running any of them.

**If you close the command window any time after running the reset path batch, you may have to re-run it each time you open a new command window.** So try not to close the current command window unless absolutely necessary. Obviously, you can develop your own testing process if you wish not to use the provided batch files. In fact, it would probably be better for you to design your own.

In the **SOFTWARE** directory, which is created or updated after you run any of the executables, are sample “freeware/shareware” programs to test for cataloging, hashing, copying, zipping/restore. If available, test your own already installed software first. If I inadvertently placed any of my licensed versions there. Please delete them, it is an error.

The **ARTICLES** directory contains the htm articles which should be reviewed to obtain an understanding as to why this session was designed. It also contains this and other documents which describe these processes. But this is the most inclusive document. The others are segments from this document.

The **TEST\_BATCHES** directory contains batch files which may assist in determining if your processes pass or fail certain test requirements. They are all command line batch files. The batch files are only provided for assistance if you wish to run them. They are not part of the processes. The files may need some tweaking to point the called programs and output directories to the correct locations. Test your batch programming skills.

Other batch files provided may be used once you examine and understand their uses.

**Develop and use your own** testing process to confirm the actions of the software. If not, the batch files are designed to show problems. But don’t necessarily show what the problem is. You may wish to examine the software execution closely to see why the software failed a particular test.

The **CATALOG\_EVIDENCE\ADMIN** directory contains administrative documents and the online test relating to the disk cataloging/listing process. And the **CATALOG\_EVIDENCE\FILES** directory contains the evidence which you will catalog. The other Dx and SOURCEx directories are described below.

The directories **D1, D2, D3 and D4** contain small samples of the evidence, or are empty for you to practice. These Dx folder are there for practice use in the hash, copy, zip sessions. The D folders are for fun, and can mostly be ignored unless you wish to run some preliminary testing to satisfy yourself of the process you run. Test batches do not exist to confirm these directory contents.

**SOURCE1 and SOURCE2 are the important directories to consider for the hash, copy, zip/unzip segments and are your “ORIGINAL” evidence to use in the testing processes**. **SOURCE1** is original; **SOURCE2** is a confirmed identical work copy of **SOURCE1**.

It is necessary that **SOURCE1, SOURCE2, SOURCE3, and SOURCE4** directories are all in the top level of the working directory or thumb drive. Review the provided confirmation batch files to make certain any paths to the data or software are properly set out. But you have designed your own confirmation process, so you don’t need mine. **YES/NO**?

**SOURCE1 is the directory containing a rescue set of data** for when you corrupt the copy of the data in SOURCE2 during the tests. Betcha: At some point SOURCE2 will need reconstructing.

**SOURCE1 should not** under normal process be touched in any way for fear of corrupting your original evidence. SOURCE1 is used by the batch files, and should be of no interest to you, other than the knowledge it is original and correct as first found on the suspect device, and properly recovered. Use SOURCE1 only to restore the data to SOURCE2 when SOURCE2 gets corrupted. Or you could just extract out the executables all over again to refresh the SOURCE directories.

**SOURCE2** (is the key) should be considered the original evidence available. Whether the suite extracted it for you, someone else obtained it, or you are sitting at the original computer holding the evidence. It is your ORIGINAL evidence to process and preserve.

*If you prefer, consider it the original subject/suspect location of forensic data to be processed (hashed, copied, zipped).* It has been certified as complete and correct original evidence. It is evidence sitting on the suspect computer for you to hash, copy, zip. (The SOURCE directories are not involved in the cataloging/listing phase. That is what the CATALOG\_EVIDENCE\FILES directory is for.)

It (**SOURCE2**) is the “sourceX” folder from which you will perform your “forensic” hashing, copying, and zip retention/restore which will create the data set that you will provide to your prosecutor.

When SOURCE2 is corrupted by your process (and it will be), then you can restore SOURCE2 from SOURCE1 which you have not touched. However, if SOURCE2 gets corrupted, how do you know any process you use to restore it from SOURCE1 will not alter SOURCE1, your original evidence? To protect you from yourself, the batch files called: **\_RESET\_SOURCE1.BAT** and **\_RESET\_SOURCE2.BAT** are provided to “reset” SOURCE1 or SOURCE2 to original settings. **\_RESET\_SOURCE2.BAT** is your safety net and should be used to restore SOURCE2 to its original evidentiary state when necessary. *Make sure the software paths called in the \_RESET batch files are all correctly updated to fit your file structure.*

**SOURCE[3,4]** will be the destination folders for the copying, and zip/restore functions of your process. SOURCE3 and SOURCE4 are currently empty folders, if **NOT**, empty them before restoring or copying any of the original evidence to them. In preparation for the tests, I may have placed a dummy file in each (SOURCE3,4) as a place holder. You can delete those before any testing. **This is where you should drop your file copies and unzipped files during the tests.**

Before running any tests you should run the **PRIMARY\_TEST.BAT** file found in the top level of the extracted data. It should run and elicit NO errors or mismatches of the files in SOURCE1 and SOURCE2. If it shows mismatches, then there is a problem it probably means that your process corrupted something along the way. Find out what/how you corrupted the data, and then run the **\_RESET\_SOURCE[1,2].bat** files to hopefully correct any problems. Use the \_RESET..bat(s) to fix SOURCE2 when necessary, then run PRIMARY\_TEST to confirm all is well. (The above process ONLY works if you are running from the root of the thumb)

The batch files such as: **TEST\_SOURCE[2,3,4].BAT** are batch files used to confirm if your hash, copy, zip/restore process was successful. As an initial confirmation, before you do anything else, you should run **PRIMARY\_TEST.bat,** and it will show that there is currently no difference between the reference base SOURCE1 and SOURCE2. This verifies that SOURCE1 and SOURCE2 are identical, and you will use SOURCE2 as your primary evidentiary source of data for the tests. SOURCE1 is a **safety net**, not to be touched. It is used by the \_RESET bat files when needed.

**ZCLEANUP.BAT** should be used to remove any temporary files that the PRIMARY\_TEST.bat or TESTxxx.bat files created when confirming your process. Run this after reviewing results of each TESTxxx.bat run. It should clean out any output files in the OUTPUT directory and make ready for the next step.

In the SOFTWARE\MWARE directory are two shell spreadsheets. You can use these and fill in the appropriate columns relating to the software you are testing. Depending on the version of the session: these spreadsheets may also show up in the hash and copy software sub-directories.

**SOURCE1 is recovery safety set. Don’t touch.**

**SOURCE2 is your original evidence.**

**DON’T touch SOURCE1.**

**CATALOG\_EVIDENCE\FILES contains files for the cataloging/listing processes**

All other folders contain administrative type items and files.

**VERY VERY VERY Important: first thing to do.**

Before performing any tests, make sure your path is properly adjusted to point to the SOFTWARE\MWARE directory, and any software or other path references in any of the batch files are properly adjusted to point to the proper location (ie: OUTPUTS, RECOVERY\_TEST, TEST\_BATCHES, etc. or any process on the data files, adjust the paths referenced in the file: **PRIMARY\_TEST.bat** and run the program.

This batch file calls (if you reset the path properly) appropriate files to test and confirm that all the files in **SOURCE1** and **SOURCE2** are identical and have not been corrupted. (note: in some cases, an additional file is contained in source2 just to make sure the PRIMARY\_TEST.BAT process I provide works).

You can use this batch, or another called **TEST\_SOURCE[2,3,4].BAT** batch files to confirm that the data in SOURCE2, 3, or 4 are identical to that in SOURCE1. If you get errors, then somewhere your process corrupted one or all of the data folders and you mist fix the SOURCEs before continuing. These test batch files, (unless you have developed your own confirmation tests) are the only way you can determine if your process worked.

=====================

Goto next page:

**Theory Behind These Test Requirements**

You will be testing a number of items in your forensic processes.

Depending on the software capability you will confirm or deny it passes the appropriate requirements.

These forensic requirements may be of different opinions based on your needs and day to day operations, the file system encountered (NTFS, LINUX ext file system, MAC, ExFat etc), and what OS is in play. Let’s just assume at this point all this evidence we are testing is found on a Windows NTFS system. The reason being is that many of the tests require the use of some of the NTFS capabilities and your software to follow such items.

Consider how a savvy defense attorney might show you haven’t tested the capabilities of your software in this situation. It could be a bad day at black rock or in court for you and be detrimental to your case. Try explaining how/why your software altered or missed exculpatory evidence.

These are the items or parameters that you will test for.

I tested for these failures, and found many software programs (over 90% of the over 40 software programs tested) will fail in one or more, and ultimately may make your arguments/evidence unsustainable.

The following topics will be primary requirements for the tests. You will be responsible for developing the process to perform the action, and then TEST to see if your software action properly handled the requirement.

* 1. The maintaining **original file dates** on **source** and destination when
  + cataloging/listing the contents of the evidence tree/directory. Find **ALL** the files.
  + hashing. (Hashing is used only to use a process which will access file content.)
  + copying. Original evidence on the thumb drive to a work location (source3) of your choice.
  + zip and unzipping evidence for retention and later review. (zipping/unzipping is copy also)
* 2. Long filenames (>255 characters) in the Windows environment, and need to properly maintain the path/filename. You will be surprised.
* 3. Alternate Data Streams (ADS) in the Windows environment and the need to properly handle them.
  + A routine action that may cause ADS’s to be created. (For instance, did you know that Firefox creates alternate data streams showing the source URL of some files when saved to disk. Other browsers save differing information in the ADS. All may be evidence or not.)
* 4. Properly copy, restore and unzip any zipped files to a similar relative tree structure.
  + Simply copy the original evidence to your work (source3) location for analysis.
  + For purposes of these tests, the zipping process is also considered a “copy” procedure.
  + Zip/restore used for evidence preservation, and delivery (to source4) to defense.

Remember, the data provided in the identified folders of the extracted data is considered **original evidence**. You should be able to see what your software does or doesn’t do in relation to processing this evidence in a forensically sound manner. The tests were set up to see how many “stand alone” packages we can break. Suites generally perform OK at the image level, but not so well at the tree level. But you can still test your suite, as you may find that some of its process will alter, or not properly restore evidence. I will tell you that the source1 (and its sister, source2) have over 125 files in them. If I told you the exact number where would be the fun.

**First item to consider: (create a complete list of the evidence tree)**

Using your software or any of the software provided in the SOFTWARE\CATALOGING directory find out how many files are in the **CATALOG\_EVIDENCE\FILES** directory.

Also, in the **CATALOG\_EVIDENCE\ADMIN** directory is a document: DiskCat\_Practical\_Student.docx which will go over the operation of the Maresware diskcat program, and show how it can be used to find various file parameters. (but use it only after you have tested your own file listing software).

Then, once you have found a reliable (ha ha) file listing package, run the **CATALOG\_EVIDENCE\ADMIN\CATALOG\_EXAM.EXE** from the admin directory. It asks about 20 questions relating to the files in the FILES evidentiary folder. Then at the end, are 3 or 4 fun questions. It creates a final output file called: student with your test answers.

After you have found a reliable file cataloging/listing software package, we will go on to the rest of the requirements in hours 3 & 4. But these requirements will be tested and reviewed in hour 3 and 4 depending on the session schedule.

**Second item to consider: (maintain original last access date of the source)**

**Last Access altered**:

For all the tests, see if the process, (hash, copy, zip/restore) alters the **original** and/or destination last access date. In addition, when copying or restoring the zip, make sure **ALL** original source and destination (MAC) dates are set or (re)set to original and properly maintained. Some programs fail to restore **all** dates and it would be embarrassing if you had to testify why your evidence process couldn’t/didn’t restore original dates.

If any of these actions alter and do not reset the last access date of the source, or do not properly reset ALL the MAC dates in the destination, it could be argued by the defense that you altered evidence, or at a later time, could not show the original dates. Suppose (for these tests and in real life) for whatever reason you can’t perform a bit image, and are forced to work on the original server directories. You can’t allow your process to alter the original dates. The owner of the hardware has: **Registry Last Access Update turned on.**

For instance, suppose you are looking to see when a file was copied (for theft of data) from the original sensitive location to another. Normally last access of a computer is turned off and this process would be difficult to track. If the corporation/custodian had last access turned on, almost any activity on the file would alter the last access. This time of last access might prove your case, or lead you to more valuable evidence. Maintaining last access date of the evidence might be important. Altering the original last access of the original evidence (**SOURCE2** folder of the test data) might be hard to explain when you try to convince defense that the file was accessed or copied on X, when your copy shows Y. Splain the difference to me.

1. Does your (first test) hashing software alter the **last access** date of the evidentiary file when it processes the file? *Again, we suggest hashing because it is a common practice in forensic validation.* And a complete hash list: isn’t it also an inventory/catalog of the files?

2. When performing a “forensic” copy, or restoring the zip contents, does the process reset/restore the destination file to the original MAC dates, or does it “set” the restored MAC date to the current time this restore/copy operation was performed? Evidence corruption?

Test to see if the MAC dates, especially original and destination last access was altered by your hash, copy, or zip and restore. If not properly restored during the copy or zip restorations it may be important to explain in testimony and evidence preservation later on down the road.

*Before you get your knickers in a wad about the fact that last access date (or any date) is easily manipulated, or it depends on the file system, or the OS being used or the phase of the moon. Yes, you are correct. But for the sake of these tests, and for possible discovery or leads to possible evidence, or defense arguments, let’s just assume for this time and these tests, that last access is important, needed, and useful in leading to valid evidentiary data.* ***No more said. Read on****.*

**Third item to consider during the test(s): (Long filenames)**

What comprises a long filename?

Filenames/paths longer than 255 characters.

Why should your software be able to handle any long filename?

Users and processes often create long filenames, in excess of 255 characters, equal to length of “War and Peace”. When a forensic suite “restores” an evidence file to your work location, it has no problem placing the restored evidence in a path with length greater than 255 characters.

In all the tests (hash, copy, zip) you perform, see that all the programs can find and process files with long filenames > 255 characters. In my testing, some programs saw them but could not access them. The programs get error messages, but no answer on how to fix the problem. Others; restored the files but only restored their 8.3 filename. Others simply ignored those files and didn’t even advise they were there. Great idea for original copy off a server to recreate an 8.3 path. A defense question. What other evidence did you alter or miss?

**Fourth item to consider: (Alternate Data Streams: ADS)**

What and where are they?

* I call them hitchhikers. X-Ways calls them child objects. Windows Explorer says DAH!
* They are hidden from “normal” view and can be or hold important information.

How to find them?

* Very carefully. Using software specially written to identify them.
* Various suites handle this process in their own way. Some even ignore ADS identification.
* Most of the stand-alone programs totally ignore them. (missed evidence???)

Can they be important?

* They can hold passwords, be the source of a pornographic image downloaded from the net, include virus software (yes executables), or just about anything you want to hide from normal processing.

Can they be hashed/copied/exported?

* Depends on the capability of your hash, copy or zip software.
* Did you ever suspect that when you perform a “forensic” copy of data from a source that you can’t BIT image that you may be leaving important evidentiary ADS files behind? This may be of particular concern when copying selected folders from a server or large data array. What did you leave behind?

Sample display (of MARESWARE – MDIR program) of an ADS file name. (With its “parent”)

**STREAM\_FILE.TXT 48|01/01/2019|12:30:01c|01/01/2019|12:30:01w|01/01/2019|12:34:56:789a|GMT|A.....**

**STREAM\_FILE.TXT:ALTERNATE.TXT 34|01/01/2019|12:30:01c|01/01/2019|12:30:01w|01/01/2019|12:34:56:789a|GMT|.adata**

My computer when the file times were set is in eastern US, and summer offset is UTC-5, but the program has the capability of showing times in GMT format. Good for consistency. (fun fact: look at the time)

With all the hash, copy, zip/restore processes, make sure the programs can see and properly process, hash, copy, zip/restore any alternate data streams associated with the evidence. You may be surprised what evidence your tools might be missing. Some programs can’t handle ADS’s at all. Others process them in weird undesirable evidentiary ways. Some process, but don’t advise they are ADS’s. Know your program.

For instance, did you know that Firefox creates alternate data streams showing the source URL of some files when saved to disk? This might be important in a pornography or virus case. URL source of the download.

Alternate data streams can contain a lot of hidden evidence that may be missed if they are not copied, retained, or properly restored. Also, when you download the NIST NSRL data sets, it forwards you to a non-NIST cloud site for the download? This download site is in an Alternate Data Stream. Try it.

If you are online and use Firefox browser: goto: dmares.com and save one of the flag images. Then see if Firefox created an Alternate Data Stream. (This expects that you have software than can show you an alternate data stream.) Windows browsers, almost always create ADS’s when saving images. But the content is different for each browser. You may not have known that. Maybe the defense or prosecutor does, and asks: where are the ADS, or where/what was the original download/source URL.

**Any, or all of the above factors (last access MAC’s not maintained, LFN’s not found or processed, ADS’ failed to copy or identify), could be ammunition for a defense. They could and should be seen as a minimal requirement where possible. That is why you will develop tests for your software, and determine if it can pass these tests. Whether you consider them important in your day-to-day forensic evidence processing, or just wish to demonstrate that software isn’t always properly advertised. Have fun.**

Case A might not require any of these items. But Case B might require them to be documented and testified to. Especially if the defense is aware of the data. And more important, does the opposition know the capability and flaws of the software you are using. Hopefully you do.

Remember, most software doesn’t contain bugs. It’s just operationally challenged ☺. And you should know or determine its challenges with relation to your process.

**Fifth item to consider: (Zip/Unzip or: Copy/Restore)**

Simply put. Can (or does) your software, whether it be a simple copy program (ie: Explorer-point and shoot copy), or a zipping program (ie: pkzip, winzip, 7-zip, Winrar, WhateverZip.exe), which is used to obtain copies of original evidence source, or zip the data for future retention, and then place it in the destination directory, or unzip it properly for future review.

Some “suites” I have tested have the capability of collecting original evidence and placing it in a “zip” container for transfer to the forensic analyst. However, this process may miss some evidence. Test your forensic suite in the “zip/compress/save” original source folder for transfer to the analyst.

Test your software to see if can take a source folderA (source2), place it into some sort of container, (zip, rar, E01, L01, Zxx) and then restore this to folder (source3), to a new location with all original evidence as it was then, is now and shall forever be identical to the original. **NO BIT IMAGING ALLOWED**. Just use the folder capture options of your suite. Bit images usually work. But do they?

The required process as explained below, will ask you to copy original suspect evidence from the **SOURCE2** directory, and copy it to the **SOURCE3** work folder, or to zip it from the original evidence **SOURCE2** and later unzip it for review into **SOURCE4** folder.

Each and every time you perform a copy, or zip/unzip you must create a testing process that will confirm or deny that your copy is complete, and has maintained and not altered any of the original evidence in SOURCE2, and properly restored it to the destination (source3 or source4).

Not as easy as it sounds.

*If, for any reason, HA HA, you corrupt the data in SOURCE2, SOURCE1 is your safety net, and you should devise a process to restore SOURCE2 from SOURCE1 for testing purposes when you corrupt SOURCE2.* ***AND IT WILL GET CORRUPTED.***

==========================================

Next page:

**Test Requirements:**

**1. Accurate file listing/catalog**

Your software that creates a catalog/list of the files in the evidence folders must be able to provide a true, accurate, and complete list of the files in the tree.

Is should also be able to provide an output, possibly clean text which can be further massaged and used in reports or spreadsheets as needed to provide to the reviewers.

Your reviewer, client may require some sort of list of the files you are providing to them as evidence.

**2. File Dates**:

* The three file dates are discussed. Modified, Created, Last Access. The registry key which determines whether the last access is updated for a file or not is discussed.
  + For adequate results, analysts **MUST** set their registry to UPDATE last access.
  + **Key Name: HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem**
  + **Name: NtfsDisableLastAccessUpdate**
  + **Type: REG\_DWORD**
  + **Value: 1 (A value of 1 turns last access update off.)**
  + **Value: 0 (Set last access update on. Access date is updated)**
* Possible arguments why you would wish to maintain ALL the file dates during your analysis.
* Do you know that a lot of virus checkers alter last access. Does this corrupt evidence time line.
* Assuming the data on the thumb is original. Do you really want to alter its dates?

Sample file with three dates you will see on the drive

**Name Size Created Modied Last Access TZ**

Lec 11.htm 52,219 01/01/2019 12:34:56c 01/01/2019 12:34:56w **01/01/2019 12:34:56a** GMT A.....

**3. Long file names:**

* What comprises a long filename?
  + Users and processes often create long filenames, equal to length of “War and Peace”
* Why should your software be able to handle any long filename?
  + Because many forensic suites will easily export evidence to long filename paths.
* How does your software handle long filenames?
  + Ignore completely
  + Change from long filename to 8.3

Sample (shortened for display) of long filename. This path is actually over 255 characters

**D:\D1\top\first\second\third\...\fourth\fifth\writ\_sec.htm**

**4. Alternate Data Streams (ADS):**

* What are they?
* Where are they?
* How to find them?
* Can they be important?

Sample directory display of an ADS file. (With its “parent”)

STREAM\_FILE.TXT 48 01/01/2020 12:34:56c 01/01/2020 12:34:56w 01/01/2020 12:34:56a GMT A.....

STREAM\_FILE.TXT:**ALTERNATE.TXT** 34 01/01/2020 12:34:56c 01/01/2020 12:34:56w 01/01/2020 12:34:56a GMT .adata

Next are the actual processes you will be performing.

**The Actual Processes**

==================================================

**FIRST Test: File cataloging/listing (hour 2)**

Before doing anything else, you probably want to know how many files are in the **CATALOG\_EVIDENCE/FILES** tree, or the **SOURCEx** directories. To do this you have to find a reliable program that will find and list or create a reliable catalog of all the files within the target environment.

Also, when you have completed your analysis, it might be nice to be able to create a similar list for retention, and one to provide to the opposition if necessary.

A good file cataloging program should be able to perform these tasks without altering the data, and provide an accurate and complete list of the files within the test area. That is why cataloging is the first test.

The **CATALOG\_EVIDENCE\FILES** folder(s) contain sample suspect directories, and other items which will help you determine if your cataloging software produces a complete and accurate listing.

Within the **CATALOG\_EVIDENCE\ADMIN** folder, is a document which contains questions and a test that you will answer once you have started using any file cataloging software you may have available. Only use the Maresware diskcat.exe program as a last result. Use your own cataloging process and see how it performs.

==================================================

From now on, you will be using the SOURCEx folders, and the Dx folders. The CATALOG\_EVIDENCE tree is not part of these next (hash, copy, zip) tests.

**HASH TEST** (hour 3)

**Test: Forensic Hashing/Hashes:**

What is a hash?

* A mathematical calculation performed on every bit of data.
* Evaluates to a unique number (often referred to as the fingerprint) representing the “content” of a file.
* This calculation does not take into account any meta data such as file date, name, size, etc.
* It only concerns itself with content.
* Used to confirm **integrity** of data. Both original and any evidentiary copies.

Types of hashes

* MD5 (most common).
  + Generates a 128 bit number indicating uniqueness (fingerprint) of the file.
  + Chances are 1 in 1038 (10 with 38 zeros) that no two files with different content will have the same hash. (*Collisions are possible, but not practical*)
* SHA. (A number of different variants, from 128 bit to 512 bit). Most common is 160 bit. SHA-2

When to hash in a forensic or evidentiary environment. (drives, files, passwords, whatever)

* The process is to first hash the source evidence (folder, server, etc). This provides a number ABCD
* Then perform the copy process to your work environment/drive/computer.
  + Especially when copying large files over a network. Hopefully no data corruption.
* Then hash the copied file which provides a number ABCD
* If the two compare, you have a good copy.
  + If not, the copy process failed in some way.
  + Maybe you didn’t copy all the files.
  + Maybe there was a computer memory failure during copy or hash. (a lot of possibilities)
* Hashes prove to the reviewer/court/attorney, etc. that the original evidence has not been altered.

Sample hash of a file

Name MD5 HASH SIZE DATE

**C:\RESTORE2.BAT 4CB29E926EFF0C184E7A5098F5FED8C2 74 01/01/2020 12:34:56w GMT**

There are about 150+- test files to process in the evidence folder (**SOURCE2**) on the drive. Devise a process to hash the files, and determine if any of the file meta data, specifically last access date was modified.

**In this step, perform a hash of all the files in the SOURCE2 Folder. This HASHing test is used solely to show or determine if the hashing can find any/all files, and/or alters the last access date. The hash values are of no concern (unless you wish to save and compare them later during the copy session).**

Obviously, before running the test(s), you should devise a way of determining exactly how many files are in your original evidence SOURCE2 folder, so that your processing logs can verify that all the files were processed. That process alone, of finding a program that can properly enumerate (count) how many files are to be processed might be your first challenge. I will tell you there are over 150 files. But the exact count is for you to figure out. Part of the forensic process is to determine amount of evidence there is. Yes/No?

Just because I’m a nice guy. Here are the stats for the D1 directory. So you can determine in advance if your software can at least agree with this count. (Note: your version may be 1 or 2 different depending on which setup you have)

7 directories, 62 files, 2,080,284 bytes, 2.08 MB

Drive Freespace 340,431,806,464

(Includes 23 Alternate Data Streams )

===

That means: 62-23=39 visible files

And of those 62 total files, here is the LFN count

**28** files, 1,010,548 bytes, 1.01 MB

(Includes **9** Alternate Data Streams) === **19** real long filename visible files

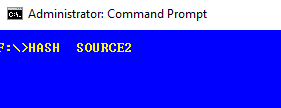
Most file listing software, including the suites don’t even attempt to count the Alternate Data Streams. Which is one of the test requirements. There could be a lot of evidence found within them. So don’t discount that data stream file count. Again, in most cases you will probably ignor ADS’s. But will you do it all the time???

This hash test to verify last access date corruption (or lack thereof) will also get you used to using the verification batch files. (*Note, for you millennials: a batch file is a script*)

Any hashing program that opens and processes a data file could be used for this test. I chose hashing because it is a very common practice and hashing programs abound. Especially when you ask for help from a list. You will be amazed at the number of recommendations or suggestions from persons who probably never really tested the software.

======================================

A simple generic command line hash test might look like this (below). But if necessary, install and run as many GUI or command line packages you have available. Some of the TEST\_BATCHES scripts have sample command lines for the associated program which I have not provided. So you may have to find and download those mentioned in some of the test demo batches.



What program(s) will you be using in addition to your own hashing package? Note: that if you have a suite that can hash files, feel free to try it along with any stand alones you have available. You can create an output or not. Just don’t place the output in any of the SOURCEx directories, as that is considered corruption in the verification steps. Create separate output locations (ie: OUTPUTS\xxx) if needed for logs, etc.

*Again: This is a good time for you to determine the exact number of files in the SOURCE2 directory. Find the correct count and remember it for later processes and court testimony. I will tell you that you should find over 150 files in the SOURCEx directory.*

Using an output file or log should reveal how many files there are. **Your hash process should not affect last access date, and it should process ALL the files, which may include long filenames, and alternate data streams**.

That is what this step is ultimately testing. How many evidentiary file items were processed by your software, and did the last access get altered? If not all, you may have missed a valuable file. Remember, in some corporate environments, you have to work on the original data because corporate policy won’t let you do otherwise.

After the hash of SOURCE2 is completed. Run the **PRIMARY\_TEST.bat** or **TEST\_SOURCE2.bat** file which should elicit NO mismatches. **PRIMARY\_TEST.bat** is preferred.

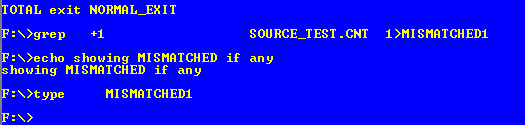
If your process altered the **last access date** of the SOURCE2 file(s) you will see a significant amount of error data at the end of the batch run. This batch file merely compares the MAC dates between **SOURCE1** and **SOURCE2**. If your hash altered SOURCE2 last access it will show up.

If you **DID NOT** hash all the files, that error **will not** show up. If you used an output log, maybe total file count would reveal if you missed some hashing. Hopefully by now, you know how many files should be processed. After all, your investigation should be able to determine how many evidence files are in play. Might be a nice number to include in your forensic reports.

Below are simple screen shots of the test batch file if NO access date alteration took place.



The “type” command as part of the batch displays 0 mismatches. All is well.



Notice when the type command was executed in the batch file, no items were displayed from MISMATCH1 indicating no mismatches were found between the current SOURCE2, and the original evidence in SOURCE1.

If **MISMATCHx** has data in it, (I may have seeded SOURCE2 with 1 additional file so you will see some errors): Open the OUTPUTS\MISMATCHx file with a text editor. The records are extremely wide. So use an editor that can sensibly display the data for easy understanding.

Once you open the MISMATCHx file, take a look at the last column. It will probably contain the value of 1. This means that the primary\_test.bat or test\_source2.bat found ONLY 1 instance of that record. Finding only one instance means that something in the record was different from the data record in source1 and source2. Most notably the access dates differ because your program altered last access of the evidence. Compare the dates from the data records for SOURCE1 and SOURCE2. Most likely one or all the dates mismatch.

Find the Access Date field and visibly confirm that this is probably where the difference occurred. You will see the access date of SOURCE1 data is 01/01/2019 and the access date of SOURCE2 record is probably todays date. The time field comparison is probably not necessary, as the date already has changed. For your own reference, I set the file dates to 2019-01-01: 12:34:56:789 GMT as explained here. How did I set original dates to that specific date/time? Very Carefully!

Perform this visual examination of the **MISMATCHx** file(s) for each of the hash, copy, and zip runs. The MISMATCHed data records is your key to finding out what part of the data record is altered. If you wish, and this should be something you design, set up your own test validation process. Knowing full well that most hashing programs under the current test environment will alter the access date, or actually miss some of the evidence data files. *Missing a file will only reveal itself if you create an output log to determine file count*.

The TEST\_SOURCE2.bat and **PRIMARY\_TEST.bat** produces the same outcome. TEST\_SOURCE2 merely presents to the user a multi-screen informational output (cumbersome).

A file MISMATCHx will be created in the OUTPUTS directory containing items that mismatch in the SOURCE2. In this case, a MISMATCH1 was created if any of the **access dates** were altered in SOURCE2. Since the hashing does not make copies, only its activity on the SOURCE2 data is tested in this step. If a log was created, see if it contains the required number of items. This batch file test only tests MAC date corruption, not file count or hash value.

If you have mismatches, then your hashing program probably altered some access dates in SOURCE2 files. Many investigations to create a baseline, may hash original evidence before copying. If so, you might be altering evidence. In many cases this last access alteration is either ignored or unknown by investigators. But a true forensic process should not alter any of the time values, or reset the time of the hash run to the current time.

The second part of the hash testing scenario is to figure out how your hashing program creates its output values. Then, create an output file and see if it in fact hashed all the test data files. A significant number of hash programs will miss many of the test data files.

**IT IS UP TO YOU, AS A SEASONED FORENSIC INVESTIGATOR TO SET UP A PROCESS TO DETERMINE IF ALL THE FILES IN THE TEST DIRECTORIES WERE HASHED, AND IF ANY OF THE FILE DATES WERE ALTERED. THIS IS THE PRIMARY REQUIREMENT OF THE HASH TESTING.**

Find the problem before continuing. Before rerunning the next hash program, don’t forget to run **\_RESET\_SOURCE2.BAT** to fix the problems in SOURCE2 that your hash run may have created.

Then confirm the restore worked by running the **PRIMARY\_TEST.BAT** which will confirm or deny (like my government speak) that your restoration was successful. No need to try subsequent tests before you fix any corrupted evidence in SOURCE2.

Run as many hashing programs as you have available to determine which files actually hash all the appropriate evidence, and don’t alter any of the MAC dates. There is a directory in SOFTWARE called: **FORENSIC\_HASHING** which contains a lot of sample hashing software I have downloaded. These are generally free packages which I have seen recommended over and over as valid hashing software. So feel free to install and test some of them. They uninstall easily.

**Be sure to run the PRIMARY\_TEST.bat or TEST\_SOURCE2.BAT after each run to see if you corrupted the dates. If so, run the \_RESET\_SOURCEx.bat batch to fix things before the next test run, and ZCLEANUP.bat to remove temp files created by the testing batch runs.**

Now that you have hashed **ALL** the files and (hopefully) found no problems, and have successfully found a hashing program worthy to be called forensic. Proceed to the copy section to determine if you have a copy program that can copy **SOURCE2** to **SOURCE3** without any errors. It should copy ALL the files without error, based on the number you identified earlier as the true number of files.

It might be helpful to create a spreadsheet to record your successes or failures with the following headings: Adjust headings based on appropriate operation and related item. Or you can use the shell provided in the root of the SOFTWARE directory.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **NAME** | **VER** | **GUI/CMD** | **UNICODE name** | **LFN** | **ADS** | **Reset SOURCE Access** | **Reset Dest.**  **Access** | **[MW]A reset dest** | **Verify Copy** |

The [MW]A reset destination column should reflect what MAC dates were properly reset to original.

==================================================

**Test: Forensically Copy the File/evidence**

Once you have determined that your hashing process didn’t alter the original file dates, and could in fact find and hash all original evidence files, the next step is to forensically copy the files from the evidence location (**SOURCE2**) to a work folder (**SOURCE3**), or for distribution to a reviewer. (Remember SOURCE2 is original evidence, possibly on a server and you don’t want to corrupt it (the original data) or the copy.)

Consider the files in SOURCE2 your original evidence. (However it was provided).

Now you must forensically copy it to SOURCE3 for additional analysis or some other production step.

If you alter SOURCE2 in any way, you have corrupted original evidence. The only way to fix it for these tests is to restore SOURCE2 as it was originally. One way is to run the batch: **\_RESET\_SOURCE2.bat**

There are folders available which will act as **source and destination** location of the copy process. (SOURCE2==source, SOURCE [3 and 4]==destination). If 3 and 4 don’t exist, make them. If they contain a few place holder files, **delete the place holder** files before doing the copy.

Once copied to the destination folder 3 or 4, you should be able to confirm

* That **ALL** the files were copied.
* That original file dates on **both** source and destination were maintained.
* If ambitious, confirm the destination file hashes are identical to original. (not required)
* **Warning: there are some intentional (realistic) problems contained within the source directory.**

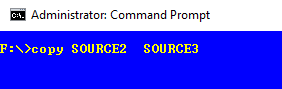
Forensically copy (this means, copy all the data, and file meta data-dates, etc) of the files in

**SOURCE2** to **SOURCE3** without altering any of the file dates, either source or destination, and make sure you get ALL the files.

After the copy, all directories: SOURCE1,2,3 should all be identical. A true copy; isn’t this what we are trying for. For the time being, ignore any MD5 matching. That can be tested at other times.

Make sure the destination folders don’t revert to their 8.3 names, which some software causes, go figure.

Generic command prompt copy command which we are attempting to find a forensic match for:



What copy program are you using? Explorer-Copy/Paste, get another job. Use your software first, and then try installing and running some of the software (**SOFTWARE\FORENSIC\_COPY\_PROGRAMS**) recommended by other smart persons. Most GUI’s need installing, but I found that they uninstall easily. (I tested over 20)

If you have a suite, such as X-Ways, FTK-Imager, Magnet-Acquire, Encase, Forensic Explorer, etc. try using that programs “copy” capability, **BUT BUT BUT**, only use the section of the suite designed to “copy” folders to containers, then extract to SOURCE3, **do not sector image the entire tree**. We know that sector imaging will perform correctly, but not always possible*. Can’t sector image a 5T server when only a single users folder will do.*

After the copy of **SOURCE2** to **SOURCE3** directory, run the TEST\_SOURCE3, TEST\_SOURCE2 or PRIMARY\_TEST.

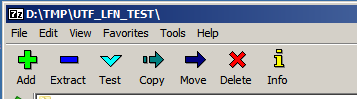
TEST\_SOURCE3 will determine if your copied data is identical to SOURCE1, original reference evidence, and TEST\_SOURCE2 will determine if SOURCE2 is identical to SOURCE1. Often SOURCE2 access date is corrupted or not completely copied or restored to SOURCE3. And your copy program may have missed some files. Find out what went wrong and Splain it to the prosecutor, defense or reviewer.

If you have any MISMATCHESx from either run, you should find out what went wrong. Remember, SOURCE1 is the original evidence which is the only reliable source to confirm no actions were corrupting the evidence. Run zcleanup to remove temporary files created by the \_TEST… programs after each run.

Once you have determined that SOURCE3 contains all the “correct” copied files you can proceed to zip SOURCE2 into a zipped file worthy of transmission to the reviewer, or for retention in the evidence safe. The next step is for zipping, storing and future retention/extraction.

==================================================

**Test: Perform a valid zip, and then unzip/restore the evidence**

 7-Zip GUI Add to or extract a zip file.

Once you have copied the files to a safe location (in this case SOURCE3), and verified the copy (**TEST\_SOURCE2, TEST\_SOURCE3**), you then should consider zipping the files for either long term evidence retention or delivery to a reviewer (i.e: attorney, manager, court, etc).

Also, for purposes of this process, the zipping process can also be considered a “sophisticated”

copy process. As some “forensic” acquire programs (such as your forensic suites) can be used to “zip”/acquire an original evidence location, which then leaves the investigator a zip type container (zip, E01, L01, x01) with which they can “hopefully” restore the original to a working location later on. I have tested one or two of these “acquire” forensic packages, and guess what, they have problems. But don’t take my word for it, I’m retired.

The zipping process should

* Properly zip all the original (**SOURCE2**) evidence, and then
* Unzip the evidence into a new (SOURCE4) folder.
  + The original zipped and unzipped files must be identical in content (hash), folder/tree layout, and file dates. (for now, ignore hash values)
  + The zipping should not alter the source meta data (ie: last access, or any MAC date)
  + The unzipped files should maintain all original file paths, file dates, alternate data streams, and any other information that could be evidentiary in the future.
  + **Basically create a full restored folder/directory tree of the original evidence**.

*REMINDER: Some programs, when unzipping, may create a significant number of top level or lower level folders within the SOURCE4 directory. Move all the lower level folders up to the correct location before attempting any of the TEST batches. They are designed to only test for the exact original tree structure of the SOURCE2 folder.*

If during the hashing, copying or zipping process the software being used cannot

* Maintain original file dates of both source and destination,
* Properly identify long filenames,
* Properly identify and copy alternate data streams.

Then your software/process may have arguable faults, should be considered as not completely forensically sound. But no software/process is perfect. They are all operationally challenged in one way or another. You must be able to justify your process. And why you accepted and ignored the incorrect operations.

Fill in the provided spreadsheets in the **SOFTWARE\MWARE** directory with the results of your tests.

It is understood, that in many environments, file dates, long filenames, alternate data streams are of no consequence. But, does an analyst know when these items will or will not be needed or challenged in court or other proceeding.

**If they become a subject of defense, can you explain why that item or process was not tested, corroborated, or maintained, and why you produced possibly altered, faulty or incomplete evidence.**

It is best to know if your software can pass these, and eventually other forensic, evidentiary tests.

If you know its shortcomings, you can (jokingly) say, I know it always does/doesn’t do this, but at least I’m always doing it the same way and not altering my process to create or miss exculpatory evidence.

Command to expand the executables is here. The passwords will be provided at appropriate time:

To extract the test data and associated items, run it whenever to completely restore the entire **NTFS** thumb.

**F:>\_EXECUTABEL\_NAME -s2 -tsp -tp+ -os –p(password provided)**

Good computing and have fun proving the software isn’t all it’s touted.

I would appreciate receiving any results you come up with.

Once again:

**Note on the extraction or download of the exe data. A few, non-main line virus checkers see the exe as a problem. Because they check for a small bit/binary sequence which they think is a virus signature. It is not. If you have a virus checker that says a virus. Use another. I’m not going to play games with faulty virus indicators.**