



---

Annual ADFSL Conference on Digital Forensics, Security and Law

2013  
Proceedings


---

Jun 11th, 2:30 PM

## Journey into Windows 8 Recovery Artifacts

W. K. Johnson  
KPMG, USA, [patories@gmail.com](mailto:patories@gmail.com)

Follow this and additional works at: <https://commons.erau.edu/adfsl>

 Part of the [Computer Engineering Commons](#), [Computer Law Commons](#), [Electrical and Computer Engineering Commons](#), [Forensic Science and Technology Commons](#), and the [Information Security Commons](#)

---

### Scholarly Commons Citation

Johnson, W. K., "Journey into Windows 8 Recovery Artifacts" (2013). *Annual ADFSL Conference on Digital Forensics, Security and Law*. 3.  
<https://commons.erau.edu/adfsl/2013/tuesday/3>

This Peer Reviewed Paper is brought to you for free and open access by the Conferences at Scholarly Commons. It has been accepted for inclusion in Annual ADFSL Conference on Digital Forensics, Security and Law by an authorized administrator of Scholarly Commons. For more information, please contact [commons@erau.edu](mailto:commons@erau.edu).

**EMBRY-RIDDLE**  
Aeronautical University™  
SCHOLARLY COMMONS

(c)ADFSL



# JOURNEY INTO WINDOWS 8 RECOVERY ARTIFACTS

W. Kenneth Johnson  
KPMG  
USA  
[patories@gmail.com](mailto:patories@gmail.com)

## ABSTRACT

One of the most difficult processes of digital forensics is to understand how new technology interacts with current technology and how digital forensic analysts can utilize current Digital Forensics technologies and processes to recover and find information hidden. Microsoft has released their new operating system Windows 8, with this new release Microsoft has added some features to the operating system that will present some interesting complications to digital forensics.

Since the initial release of the Windows 8 Release Candidates there have been some research released that focus primarily on the new user created artifacts and a few artifacts that have been added by the operating system that might contain valuable information. This paper will look at the new recovery options that have been introduced in the final release of the Windows 8, and the impact that have on the artifacts.

This paper will investigate the impact on system and user artifacts when the Windows 8 recovery methods are used. This paper will look the artifacts that are created between the different recover methods, as well as what artifacts can be recovered from the hard drive after a recovery method has been used.

**Keywords:** Windows 8, Digital Forensics, Recover Options, System Reset, System Refresh, File History

## 1. INTRODUCTION

One of the most difficult processes of digital forensics is understanding how new technology interacts with current technology and how we can utilize current Digital Forensics technologies and processes to recover and find information hidden. Microsoft has released their new Operating System Windows 8, with this new release Microsoft has added some features to the Operating System that will present some interesting complications to digital forensics.

Over the last year there has been plenty of research released that focus on the new user created artifacts and a few artifacts that have been added by the operating system that might contain valuable information. This paper will cover the new recovery options that have been introduced in Windows 8, and the impact that have on the artifacts.

The first thing that this paper will cover is the artifacts discovered by the research of Amanda Thomson. Once these artifacts have been analyzed and verified the locations on the disk, a baseline dataset will be created to compare the impact of the recovery options on these artifacts. This baseline dataset will be used for finding artifacts on the system after a recovery option has been used.

The final thing that this paper will cover is how the various recovery options impact the artifacts that are found on the operating system. This will be done by installing Windows 8 in a Virtual Machine environment and taking snapshots of a base image and then utilizing the various recovery methods. Once the recovery method has been successful ran, analysis of the Virtual Machine will be done by importing the image into FTK Imager.

## **2. GATHERING OF INFORMATION**

### **2.1 User Created Artifacts**

Amanda Thomson released documentation on various artifacts that the new Metro interface of the Windows 8 operating system creates (Thomson, Propellerhead Forensics, 2012). This paper will focus on the artifacts that are contained in the communication apps, internet explorer 10, File History and user created documents. This will create a baseline of data to search for and extract from.

### **2.2 Volume Shadow Copy Services**

Within the Microsoft Windows 8 Operating System, Microsoft has enhanced the Volume Shadow Copy Services to be utilized by more than just System Restore Points. This service is also now the backbone of the File History Service and is also utilized with the system refresh. This paper will look at the integration of the Volume Shadow Copy services and what artifacts will be useful for analysis. This paper will examine how these artifacts are impacted and used by the various recovery methods (Microsoft, n.d.).

### **2.3 System Restore Points**

System Restore Points are created three different ways within Windows 8. Like previous versions of Windows these can be created via System Initiated process, and user initiated process. Within Windows 8, two new registry values have been created that allows applications to initiate the request for systems restore point creation. This paper will utilize the Communication Application artifact discovered from Thomson's research as well as research of my own to compare between what was expected and how it can be recovered. As with previous versions of windows previous version copies are stored in the Volume Shadow Copies and can be recovered by mounting the drive and extracting the data.

### **2.4 System Refresh**

Windows 8 has introduced the ability for users to recover from malware infection or stability issues by including the refresh option in the operating system. There are two options with the System Refresh that users can utilize. The first option is a default refresh which will revert the operating system back to a factory default setting and a custom refresh that allows the user to define the snapshot scope to revert back to. In gathering information on what artifacts are impacted and retained from utilizing the System Refresh this paper will analyze both the Custom and Default against my baseline and compare the difference. This paper will also identify new artifacts that are created on the hard drive from this process.

### **2.5 System Recovery**

Windows 8 has introduced the ability for users to quickly reinstall the Operating System from a GUI for the user. The system recovery offers a few options for reinstallation; these options are Quick and Thorough Recovery. The differences between these two options are the ease of which data can be recovered. The quick recovery still allows for easy extraction of files on the machine; while the thorough recovery makes data recovery difficult. This paper will look at the impact on the baseline dataset artifacts when the recovery options are utilized and will also indicate new artifacts that are created when these are run.

## **3. FILE HISTORY SERVICES**

Within the Microsoft Windows 8 Operating System, they have introduced file history backup, which changes the way backups were previously used. In previous versions windows could only maintain and restore backups using the default system. With Windows 8, Microsoft has implemented a solution that is more robust and allows backups to be stored both on removable media and remote network

shares. By default File History will back up the following folders: Music, Documents, Videos, Pictures, Desktop, Contacts and Favorites (Serban, n.d.).

There are a few artifacts that are established when the File History is turned on these include the file history folder and registry values. The file history folder can be found in the following path: **C:\Users\<USERID>\AppData\Local\Microsoft\Windows\Filehistory**

This directory is also written to the backup location. Within in that directory there are two folders named Data, and Configuration. The data folder contains the files and folders that are tagged for backup using the file history. The configuration folder contains files that are both EDB and XML files. File names for the EDB follow the naming conventions of Catalog#.edb, and file names for the XML files are Config#.

<b>File History Config Values</b>	<b>What does it mean</b>
UserName	<b>The user account that this is configed for.</b>
Friendly Name	<b>First/Lastname tied to account. Inherited from Windows LiveID if configured</b>
PC Name	<b>Name of the computer</b>
UserType	<b>What type of user this is</b>
Library	<b>Directories that are being backed up</b>
UserFolder	<b>User Specific Directories being backed up. Sub Value of Library</b>
LocalCatalogPath	<b>Filepath to the Catalog.edb and the config files</b>
Retention Policy	<b>How long data is retained</b>
DPFrequency	<b>How often is the data backed up</b>
Target	<b>The backup location</b>
TargetName	<b>Name of the Location used for backup</b>
TargetDriveType	<b>The options Local, Remote, and Removable are for the drive type that the backup is sent to.</b>

If the File History option has been turned on, there will be registry keys created in the HKU keys of the users that have this option turned on. This key can be found in the Software\Microsoft\Windows\CurrentVersion\FileHistory. Within this directory there is a key names ProtectedUpToTime which is a 64 Bit

Hex Value–Big Endian, which can be deciphered by utilizing the DCode application. In Figure 1 the FileTime value is seen, in Figure 2 the value has been converted in the application DCode, this time represents the last time an update was pushed to the file history system.

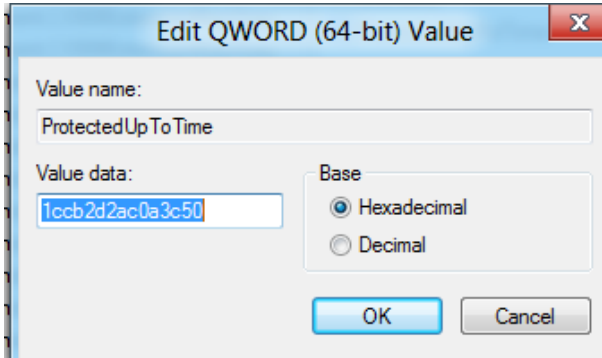


Figure 1

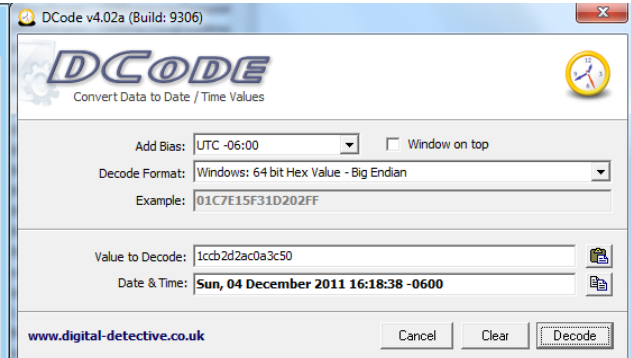


Figure 2

Another area in the registry that may contain keys of importance is HKLM\System\Controlset001\Services\fhsvc. Within this Key, there is a parameter key that shows the location of the configurations values.

Another area to look at in gathering File History information is within the System Events. The following Event Sources provide us with information related to the File History: FileHistory-Catalog, FileHistory-ConfigManager, FileHistory-Core, FileHistory-Engine, FileHistory-EventListener, and FileHistory-Service. As of this research the event logs being parsed are the ones that show errors, and one that fires off with each successful backup, but claims something is missing and can't be parsed. Until this operating system is further along, this issue might remain.

You can also utilize the Jump Lists for the File History to gather more information on it. I was able to pull from the file history jump list, the various drives I used for my backup locations. This will be beneficial if the user modifies their backup location.

If a Windows 8 machine has the File History Service turned on, it will persist over a system Refresh. These files can be found in the original directory.

#### 4. COMMUNICATION APPLICATION

Some of the more useful artifacts that are included with Windows 8 as the Metro App known as the Communication App. Amanda Thomson went into great details in her research on the artifacts that can be recovered. She showed that analysts are able to extract artifacts that will show who and how a user interacts with various online organizations and people (Thomson, Windows 8 Forensic Guide, 2012).

From within the Communication App there are few locations that will provide artifacts for analysis; Thomson touched on the Cache and the Mail locations. This paper will discuss the following artifacts that I discovered in the application; AddressBook and Me directories. While there may be more of value, these two locations open up more insight on the users contacts and the accounts being used by the user.

From the cache files analysts are able to extract the user online contacts details. These details include email, twitter handle, profile pictures, and pictures that were shared by the users contact. From the cookies files analysts can extract conversation, email, email attachments, twitter communications and other communication transactions between users.

From the cookies files analysts are able to extract user messages that have appeared in the communication app. Thomson's research showed how email messages appeared in the cookies directory, this example shows how a Twitter stream would appear in the cookies directory. In this

example we can see the username that posted to tweet, the content of the tweet and the associated url that was included in the tweet.

```
<item name="3268230390960891222_e" value=["TWITR_197188068","TWITR__WHATSNEW__
ltime="1480131536" htime="30248684" />
<item name="3268230390960891222_e_TWITR__WHATSNEW__"
value=[{"id":"244926118065487872","sourceId":"TWITR","type":1,"data":{"type":1,"publis
{"id":"19491279","sourceId":"TWITR","name":"David
Barroso","networkHandle":"lostinsecurity","picture":"http://a0.twimg.com/profile_images
screen_name=lostinsecurity&size=original","profileUrl":"http://twitter.com/lostinsecurity'
{"text":"Huawei and Intel Corp join hands for IT solutions venture http://t.co/PhFBLeI9 <-
McAfee???"},"via":"Twitter","entities":{"type":2,"data":
```

Figure 3

From the mail files analysts can recover information about the emails the sender has sent, received, stored or even ads that have populated the inbox of the live account tied to the communication app. According to Thomson's research the file path to the Mail directory is the users windows live account. This is no longer the case as it is now a random string. I have verified that by having the same windows live account across multiple machines and this 16 alpha numeric character directory does not share the same name across the machines.

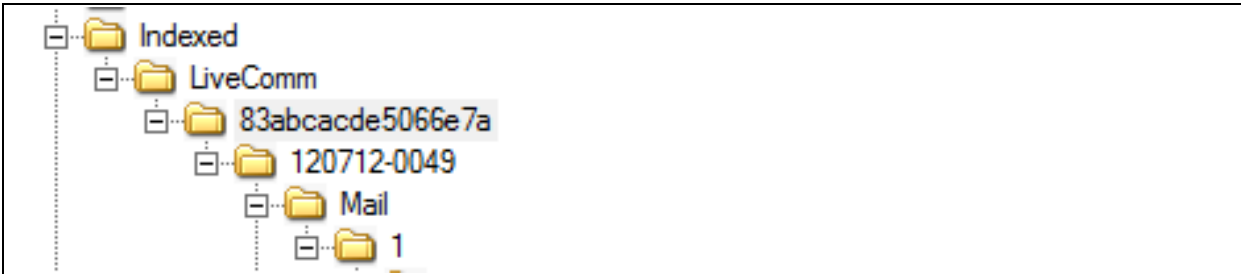


Figure 4

Within the Mail directory are subdirectories that hold various files. The subdirectories on my machines met the following naming standards 1d00000# while in Thomson's research these values were 1200000#, it appears that this naming standard follows the similar hex pattern, although at this time it is unknown what the meaning of the pattern is. The files in the subdirectories in my testing followed the naming standard of:

## 2000000#\_#####.eml.OECustomProperty

(14 alpha numeric characters)

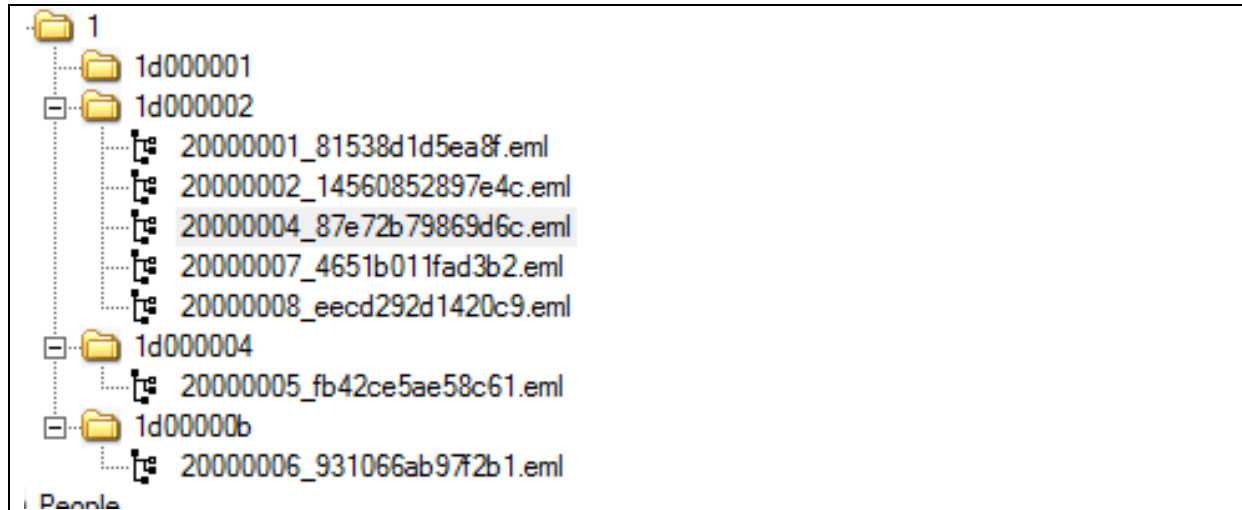


Figure 5

The different subdirectories under the mail directory appear to be different directories in the Windows Live email system. In my case directory 1d000002 appears to be my inbox, the directory 1d000004 appears to have been my sent folder, and the 1d00000b appears to be my draft folder.

From the AddressBook file analysts are able to gather username of the contacts. All entries will contain a From field which will list the Contact name, at the end of the entry there will also be a list of all alias's tied to that account. These alias can be email usernames, full names, first name, last name or even another screen name.

For a contact with an associated email the entry will contain a Subject line that will have the value HasEmail if this account is tied to an email address. There will be a unique Hex String for each email address associated with the contact in the TO: field

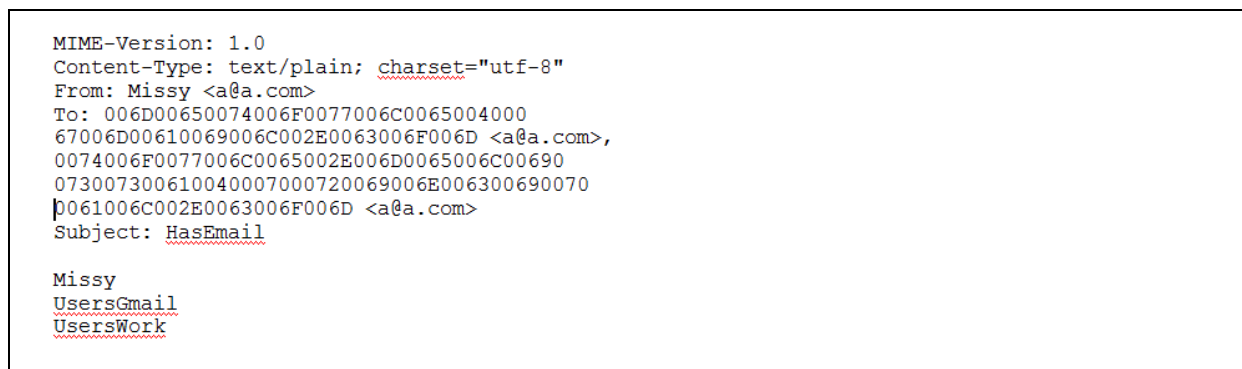


Figure 6

For a Twitter contact the two fields at the bottom of the artifact will list the screen name first and the first and last name that was entered into the product. Since this is a twitter account and no email is associated with it there is no subject that shows the HasEmail value.

```
MIME-Version: 1.0
Content-Type: text/plain; charset="utf-8"
From: VirusShare <a@a.com>

VXShare
VirusShare
```

Figure 7

For a Facebook contact the artifact will contain the full name in the from field, and at the end of the artifact the first line will again be the full name, the second line will be the first name associated with the account and the last line will be the last name associated with the account.

```
MIME-Version: 1.0
Content-Type: text/plain; charset="utf-8"
From: Add Pitch <a@a.com>

Add Pitch
Add
Pitch
```

Figure 8

The Me folder will contain an artifact entry that will contain all the accounts that the user has connected to the communication app. This artifact follows the same structure as previous examples; containing the hex string of the email address, user name and the associated full name. As shows in figure X, I used a Hex to ASCII converter to change my stored email from Hex to ASCII.

<pre>MIME-Version: 1.0 Content-Type: text/plain; charset="utf-8" From: Kenneth Johnson &lt;a@a.com&gt; To: 007000610074006F007200690065007300400067006 007000610074006F007200690065007300400067006D006 Subject: HasEmail  Kenneth Johnson Kenneth Johnson patories patories</pre> <p style="text-align: center;">Figure 9</p>	<p style="text-align: center;"><b>Hex To ASCII Converter</b></p> <p>Hex: <input type="text" value="007000610074006F0072006900"/></p> <p>Ascii: <input type="text" value="patories@gmail.com"/></p> <p style="text-align: center;">Figure 10</p>
---	---

Table 1 is an update to the artifact list which includes other artifacts be of interest to an analyst. I have included the location of the Cache and Mail artifacts as well as introducing the Adressbook and the ME locations.



Table 1

Artifact Type	Artifact Location	Purpose
Cache	%Root%\Users\%User%\AppData\Local\Packages\microsoft.windowscommunicatisapps_8wekyb3d8bbwe\AC\INetCache	Contains contacts email, screen name, or images the user has viewed.
Cookies	%Root%\Users\%User%\AppData\Local\Packages\microsoft.windowscommunicatisapps_8wekyb3d8bbwe\AC\INetCookies	Copy of messages that have shown up in the Communication App.
Mail	%Root%\Users\%User%\AppData\Local\Packages\microsoft.windowscommunicatisapps_8wekyb3d8bbwe\LocalState\Indexed\LiveComm\%randomString%\%randomString\Mail	Copy of users emails, these will contain sender, recipient, subject, body and attachments.
Address Book	%Root%\Users\%User%\AppData\Local\Packages\microsoft.windowscommunicatisapps_8wekyb3d8bbwe\LocalState\Indexed\LiveComm\%randomString%\%randomString\People\Address	Contains username and screen name of contacts. If account has email address then email address is also stored in hex value.
ME	%Root%\Users\%User%\AppData\Local\Packages\microsoft.windowscommunicatisapps_8wekyb3d8bbwe\LocalState\Indexed\LiveComm\%randomString%\%randomString\People\Me	Contains username and screen name of users accounts and all associated email addresses is also stored in hex value.

## 5. WINDOWS 8 RECOVERY ARTIFACTS

### 5.1 System Restore Points

Windows 8 maintains the traditional Restore Points that have been seen in previous versions of windows with a few new tweaks. System Restore points automatically monitors files in the boot volume that are relevant for restore only, this is incompatible with previous versions. It allows users to undo a change that may have caused a problem with the system, or to revert to a day when the system might was performing optimally (Microsoft, n.d.).

Within Windows 8 a new registry key was added that enables application developers to change the frequency of restore-point creation. If the key does not exist then when an application calls the **SRSetRestorePoint** function to create a restore point, Windows skips creating this new restore point if any restore points have been created in the last 24 hours (Microsoft, n.d.).

With this new registry key, developers can write applications that create the **DWORD** value **SystemRestorePointCreationFrequency** under the registry key **HKLM\Software\Microsoft\Windows NT\CurrentVersion\SystemRestore**. The value of this registry key can change the frequency of the restore point creation.

When the application calls **SRSetRestorPoint** to create a restore point, and the registry key value is 0, system restore does not skip creating a new restore point. If the application tries to create a restore

point, and the registry value is the integer N, than system restore skips creating a new restore point if any restore points were created in the previous N minutes.

Developers can write applications that create the **DWORD** value **ScopeSnapshots** under the registry key **HKLM\Software\Microsoft\Windows NT\CurrentVersions\SystemRestore**. If this value is 0, System Restore creates snapshots of the boot volume in the same way as earlier versions of Windows. If this value is deleted, System Restore running on Windows 8 resumes creating snapshots that monitor files in the boot volume that are relevant for the system restore only.

## 5.2 System Refresh Points

Windows 8 introduces two new options for system recovery, these options are: Refresh Points and System Recovery. Within Refresh Point there are two options; you can utilize the default refresh point or a custom refresh point.

Both Refresh options can be utilized by Windows 8 to remove malicious files and corrupted entries into the operating system. When using Refresh it is important to understand that the operating system creates a Recovery Image that makes a backup of the Windows System Files. For the default recover these Windows System Files are from when Windows 8 was first installed. When the Custom Refresh option is used than the Windows System Files are from the date that the Custom Refresh was created, the Custom Refresh also will contain the desktop applications that you have installed. Refresh Images **DO NOT** contain your Metro-style apps, documents, personal settings or user profiles, this is because that information is preserved at the time you refresh your PC.

When looking at an image of the Windows 8 operating system from within the AccessData FTK Imager there are three things that are quickly noticed. There are two partitions and an unallocated space. Figure 11 shows this layout, this is different from previous versions because of the Partition 1.

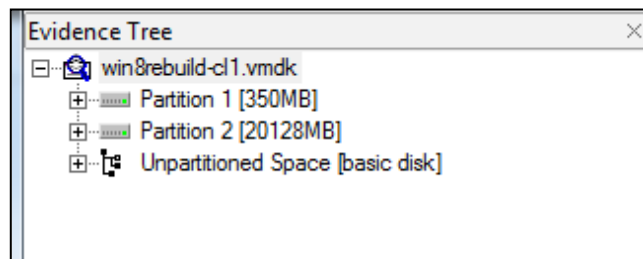


Figure 11

The Recovery Partition comes up as Partition 1 in FTK Imager and is a 350MB partition. This partition is dedicated to the basic root of the operating system. This will contain information related to the refresh process of the machine. If a Refresh or Reset has been utilized there will be a new subdirectory called Logs, which will contain a file called Reload.xml.

Figure 12 shows what the recovery partition looks like before a Refresh or Reset has been utilized. Figure 13 shows what the recovery partition looks like after a Refresh or Reset has been utilized. By looking at the two figures the difference can be seen with the addition of the LOGS subdirectory under the Recovery Directory. Within the LOGS directory there will be at least one file which will be called ReLoad.xml. There is potential to be other files only if a Refresh or Reset option has encountered any errors in the process.

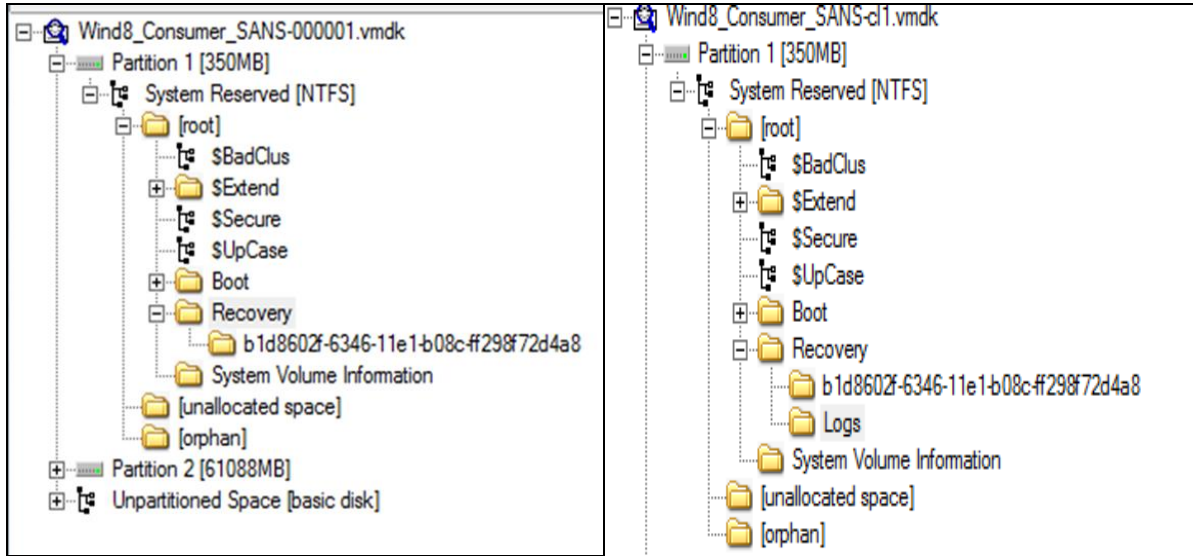


Figure 12

Figure 13

Within the subdirectories of **Logs** and the **b1d8602f-6346-11e1-b08cff298f72d4a8** there will be artifacts that can indicate if a Refresh or Reset has occurred. If the Logs subdirectory is present then the timestamp on the Reload.xml is the time and date when the Refresh or Reset was initiated. The timestamp on the ReAgent.xml which is found in the b1d8602f, indicates when the OS was successfully installed, this will update to after the timestamp on the Reload.xml.

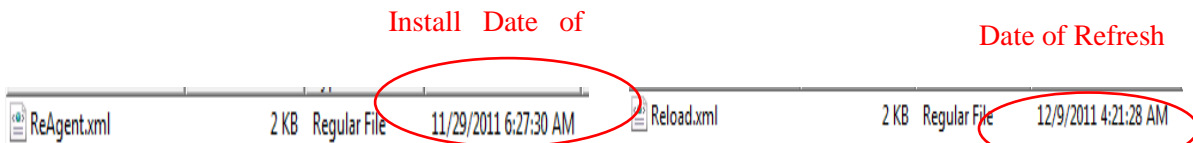


Figure 24

Figure 3

Within the ReAgent.xml we can see various configuration options for the System Refresh as well as see if it is a default or custom refresh. The first screen shot is the default refresh, the second one is from a custom refresh stored locally, and the final one is a custom refresh stored on a remote drive.

- As we can see the images share similar data between them. This includes the following:
- WinreBCD ID = is the same identifier of the folder on the system drive.
- WinreLocation Path = Where the WIMRE.wim file is found on the system drive.
- OSBuildVersion Path.

As we can see in all three images, there is a value called CustomImageLocation Path that is populated when a Custom Refresh Image is created. Figure 20 is the Reagent.xml with the default Refresh Point, while Figure 21 is the ReAgent.xml with a custom refresh location. The only apparent difference is the CustomImageLocation path is now populated with the new file path.

```
<?xml version="1.0" encoding="utf-8" ?>
- <WindowsRE version="1.0">
  <WinreBCD id="{b1d8602f-6346-11e1-b08c-ff298f72d4a8}" />
  <WinreLocation path="\Recovery\b1d8602f-6346-11e1-b08c-ff298f72d4a8"
  id="1875952216" offset="1048576" guid="{00000000-0000-0000-0000-000000000000}" />
  <ImageLocation path="" id="0" offset="0" guid="{00000000-0000-0000-0000-000000000000}" />
  <OsInstallLocation path="" id="0" offset="0" guid="{00000000-0000-0000-0000-000000000000}" index="0" />
  <CustomImageLocation path="" id="0" offset="0" guid="{00000000-0000-0000-0000-000000000000}" index="0" />
  <InstallState state="1" />
  <OsInstallAvailable state="0" />
  <CustomImageAvailable state="0" />
  <WinREStaged state="0" />
  <OperationParam path="" />
  <OsBuildVersion path="8250.0.amd64fre.winmain_win8beta.120217-1520" />
  <OemTool state="0" />
  <BootKey state="0" />
  <IsServer state="0" />
  <ScheduledOperation state="5" />
</WindowsRE>
```

Figure 16

```
<?xml version="1.0" encoding="utf-8" ?>
- <WindowsRE version="1.0">
  <WinreBCD id="{b1d8602f-6346-11e1-b08c-ff298f72d4a8}" />
  <WinreLocation path="\Recovery\b1d8602f-6346-11e1-b08c-ff298f72d4a8" id="1875952216" offset="1048576"
  guid="{00000000-0000-0000-0000-000000000000}" />
  <ImageLocation path="" id="0" offset="0" guid="{00000000-0000-0000-0000-000000000000}" />
  <OsInstallLocation path="" id="0" offset="0" guid="{00000000-0000-0000-0000-000000000000}" index="0" />
  <CustomImageLocation path="" id="0" offset="0" guid="{00000000-0000-0000-0000-000000000000}" index="0" />
  <CustomImageLocation path="\Custom_Refresh\" id="0"
  offset="1048576" guid="{00000000-0000-0000-0000-000000000000}" index="1" />
  <InstallState state="1" />
  <OsInstallAvailable state="0" />
  <CustomImageAvailable state="1" />
  <WinREStaged state="0" />
  <OperationParam path="" />
  <OsBuildVersion
  path="8250.0.amd64fre.winmain_win8beta.120217-1520" />
  <OemTool state="0" />
  <BootKey state="0" />
  <IsServer state="0" />
  <ScheduledOperation state="5" />
</WindowsRE>
```

Figure 17

To create a custom recovery image, you need to use the recimg.exe. When you create a custom recovery image, recimg will store it in the specified directory, and it is set as the active image. If a custom recovery image is set as the active recovery image, Windows will use it when you refresh your PC. All recovery images have the filename install.wim. If no install.wim file is found in the active recovery image directory, Windows will fall back to the default image.

```
C:\Windows\system32>recimg /showcurrent
\\?\GLOBALROOT\device\harddisk2\partition1\Custom_Refresh
RecImg: Operation completed successfully
```

Figure 18

```
C:\Windows\system32>recimg /showcurrent
\\?\GLOBALROOT\device\harddisk0\partition2\Custom_Refresh
RecImg: Operation completed successfully
```

Figure 49

The System Partition is identified in FTK as Partition 2. On a refresh this partition contains the following folders: root, orphan and unallocated space. It also contains the files called backup boot sector and file system slack. The root folder contains the files and settings used within the operating system. Some key information that can be found in this partition Registry Hives, File History configuration and data, contacts, documents, windows.old and other locations. Depending on how recent the refresh happened there might be a HTML file on the desktop that lists what applications are removed.

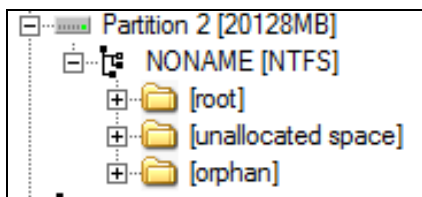


Figure 5

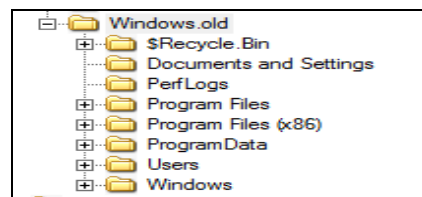


Figure 21

The windows.old folder, which is located in the root directory, holds artifacts of what files and programs were installed on the machine before the refresh. From this information it appears that on a refresh nothing is uninstalled or physically deleted from the hard drive but marked to allow the space being over written, you can still access them through devices such as FTK Imager. The files System Partition that are marked for deletion can still be accessed and analyzed as normal. I was able to pull the Registry Hive files and run them through RegRipper and Registry Decoder for data analysis.

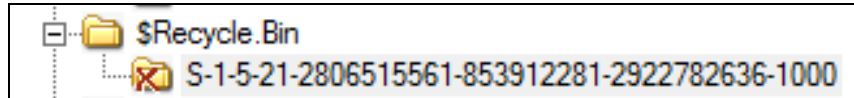


Figure 22

The Unallocated Space, contains unallocated file space, while there might be information stored there that will have value, I did not find anything in this research scope. The System Partition contains a wealth of information that should be interesting to an investigator.

### 5.2 System Reset Artifacts

When the System Reset recovery option has been utilized on a machine the data that was once available on a live system is no longer available. Depending on what type of System Reset option is used the data may not be recoverable. If a Quick System Reset is used analysts will be able to extract artifacts from the unallocated space which will include application, system and user data. If a Thorough System Reset is used than user created data will not be found in the unallocated space. Both the System Reset and System Refresh will create identical artifacts in the Recovery Partition, these artifacts will be an updated ReAgent.Xml and a Logs Directory which will contain the Reload.xml file.

If we look into the Logs file we will see the file called Reload.xml within this file has some information related the refresh/restore process. We also notice that there is a file in the system32/Recovery called ReAgent.xml, this file can be found across all three images; Base, Restore and Refresh. When comparing the ReAgent file across the three platforms the files are identical. When comparing the ReAgent and the Reload files against each other they are identical except for the following line ScheduleOperation State. The first one is ReAgent, the second is the reload file.

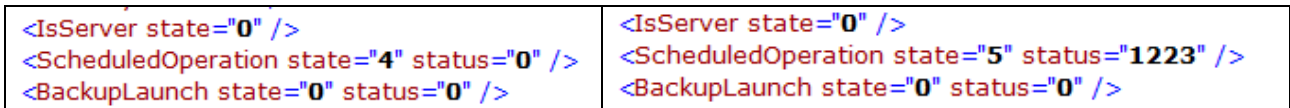


Figure 23

Figure 24

The Reload.xml file is identical across both a Restore and a Refresh process. So this value must dictate that certain files were restored to the factory default. This may change with machines that are running the Unified Extensible Firmware Interface.

Even though Windows 8, has the ability to reset the operating system back to factory default, it is not impossible to do analysis and data carving on the unallocated spaces to find artifacts that once remained on the machine. As you can see from the image below, even after I did a reset on the machine, which should have returned everything to the factory default, I was still able to carve out my test file, my username, and some files related to the file history option.

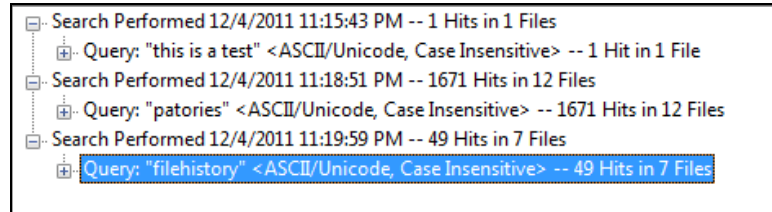


Figure 25

## 6. CONCLUSION

With the inclusion of new features in Windows 8 there is an increased volume of artifacts that will be useful to analysis for profiling user behavior and actions. While the majority of the features have made it easier for analysis some have increased the difficulty for extraction.

One of the more Forensic friendly features added to Windows 8 would be the Communication App. This app if utilized by the end user is able to provide a wealth of Forensic artifacts that might normally be available on a users machine. Through the artifacts that are created I am able to see the users various contacts across all connected accounts as well as messages, usernames and email addresses. This application allows analysis to get information that at times required contacting individual third parties for the data.

With the addition of FileHistory in Windows 8 it has given analysts potentially new avenues for investigation. With the ability to have many accounts backing up to the same location analysts now have the ability to find outlying computers or accounts that were previously unknown. With the behavior of changing FileHistory locations not clearing data from previous locations this gives analysts the ability to find evidence on forgotten drives.

Finally the availability of an interactive GUI for system recovery allows end users to quickly restore their system to a previous state. Depending on the recovery method utilized on the system it would be beneficial for analysts to know which method result in the destruction of artifacts before spending time trying to analyze a system that has been reset.

Except for the loss of artifacts from a Thorough System Reset, Microsoft has made the ability to find forensic artifacts easier. With the multiple locations that an artifact may remain in, the ease of backing up data, and the importation of third party account information the amount of user identifiable information is greatly increased.

## REFERENCES

- Microsoft. (n.d.). *Calling SRSetRestorePoint*. Retrieved from [http://msdn.microsoft.com/en-us/library/windows/desktop/aa378727\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/aa378727(v=vs.85).aspx) in 2012.
- Microsoft. (n.d.). *System Restore*. Retrieved from [http://msdn.microsoft.com/en-us/library/windows/desktop/dd408121\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/dd408121(v=vs.85).aspx) in 2012.
- Microsoft. (n.d.). *Volume Shadow Copy Services*. Retrieved from [http://msdn.microsoft.com/en-us/library/windows/desktop/bb968832\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/bb968832(v=vs.85).aspx) on March 16, 2013.
- Serban, A. (n.d.). *How to use the Windows 8 File History*. Retrieved from <http://www.itproportal.com/2011/11/28/how-use-windows-8-file-history-system/> on December 2, 2011.
- Thomson, A. (2012, 04 18). *Propellerhead Forensics*. Retrieved from [http://propellerheadforensics.com/2012/04/18/download-windows-8-forensic-guide/thomson\\_windows-8-forensic-guide/](http://propellerheadforensics.com/2012/04/18/download-windows-8-forensic-guide/thomson_windows-8-forensic-guide/) on January 15, 2013.

Thomson, A. (2012, May). Windows 8 Forensic Guide. Retrieved from [http://propellerheadforensics.com/2012/04/18/download-windows-8-forensic-guide/thomson\\_windows-8-forensic-guide/](http://propellerheadforensics.com/2012/04/18/download-windows-8-forensic-guide/thomson_windows-8-forensic-guide/) on July 2012.

### **ACKNOWLEDGMENTS**

I would like to thank my committee chair, Dr. Yong Guan, and my committee members, Dr. Doug Jacobson, and Dr. Jennifer Davidson, for their guidance and support throughout the course of this research.

In addition I would also like to thank my family for their encouragement, to my wife, Jessica, and children, Savannah and Brady, their hours of patience, respect and love. Thank you for believing in me and pushing me to succeed.

Finally, thanks to Leocadia Adams for your encouragement and always seeing the best in me. The compassion and life lessons that you taught me will always be appreciated.

### **AUTHOR BIOGRAPHIES**

Kenneth Johnson recently graduated from Iowa State University with a Master's of Science in Computer Engineering and Information Assurance. His undergraduate degree is in Information Systems Security from ITT-Tech. His primary interest is in Malware Behavioral Analysis and Reverse Engineering. He currently works for a Big 4 Auditing firm in Chicago with responsibilities in digital forensics, incident response and malware analysis.

Kenneth has presented at SANS DFIR Summit 2012, GFIRST 2012, FACT 2012, TechnoSecurity Conference 2013. He is also the organizer of the BSidesIowa Security Conferences.