Windows 10 Forensics

175 Lakeside Ave, Room 300A
Phone: (802)865-5744
Fax: (802)865-6446
http://www.lcdi.champlain.edu

4/22/2015
Disclaimer:

This document contains information based on research that has been gathered by employee(s) of The Senator Patrick Leahy Center for Digital Investigation (LCDI). The data contained in this project is submitted voluntarily and is unaudited. Every effort has been made by LCDI to assure the accuracy and reliability of the data contained in this report. However, LCDI nor any of our employees make no representation, warranty or guarantee in connection with this report and hereby expressly disclaims any liability or responsibility for loss or damage resulting from use of this data. Information in this report can be downloaded and redistributed by any person or persons. Any redistribution must maintain the LCDI logo and any references from this report must be properly annotated.
# Contents

Introduction ............................................................................................................................................................. 3
Background: ........................................................................................................................................................ 3
Purpose and Scope: ............................................................................................................................................. 3
Research Questions: ............................................................................................................................................ 3
Methodology and Methods ..................................................................................................................................... 4
   Equipment Used .................................................................................................................................................. 4
   VM Hardware .................................................................................................................................................. 4
   VM Hardware .................................................................................................................................................. 5
   Software Installed ........................................................................................................................................... 5
   Data Collection .............................................................................................................................................. 5
Analysis ................................................................................................................................................................... 6
Results ................................................................................................................................................................... 6
   Different/Updated Artifacts ............................................................................................................................ 6
      Recycle Bin ................................................................................................................................................ 6
      Thumbnails ............................................................................................................................................... 9
      OneDrive ............................................................................................................................................... 10
      Prefetch Files ....................................................................................................................................... 12
   New Artifacts ................................................................................................................................................ 13
      Spartan Browser ........................................................................................................................................ 13
      Facebook App ....................................................................................................................................... 15
   Similar/Unchanged artifacts .......................................................................................................................... 19
      Event Logs .............................................................................................................................................. 19
      Internet Explorer ................................................................................................................................. 20
      USB Activity ........................................................................................................................................ 21
      LNK Files .............................................................................................................................................. 22
Conclusion ............................................................................................................................................................ 22
Further Work ......................................................................................................................................................... 23
Acknowledgements: .............................................................................................................................................. 23
Introduction

The mission of this project is to discover differences in the artifact locations of Windows 8 and Windows 10. It will also be within the scope of this project to find and discover new artifacts that are linked to new features added to Windows 10.

Background:

At the time of writing, no prior research had been done on Windows 10 forensics. This, in addition to the lack of tools capable of performing acquisitions on Windows 10 devices, makes this project important.

Although no resources for Windows 10 exist currently, there are many resources that detail Windows 8.1 artifacts, which will be used for a comparison. Kyle Tellers, an LCDI employee, has also written a report on Windows 8.1 forensics, which will be used as a reference in this report.

Purpose and Scope:

The results of this research will be useful for forensics investigators encountering Windows 10 computers. These computers are expected to enter the consumer market in either the Summer or Fall of 2015.

Artifacts to be compared to Windows 8 in this stage of analysis are the following:

1. Event Logs
2. Internet Explorer
3. USB Activity
4. LNK Files
5. Recycle Bin
6. Thumbnails
7. OneDrive
8. Prefetch Files

New potential artifacts in Windows 10 are the following:

1) Notification Center
2) New Start Menu
3) Frequent Folders
4) Cortana
5) Synced Wi-fi Hotspots
6) Windows 10 Applications (Mail, photos, Facebook, etc.)
7) OneDrive data

Research Questions:

1) What artifact locations have changed in Windows 10?
2) What new features in Windows 10 could lead to more useful forensic artifacts?
3) Where can these new artifacts be found and how can they help a forensic investigation?
4) What artifacts can be found that are synced with other devices (OneDrive data)?
5) What artifacts can be found from common Windows 10 applications?
Methodology and Methods

The best way to analyze Windows 10 is to create a realistic investigation. For the beginning of the project it may be acceptable to export the Windows 10 registry and analyze data from the .reg file, but eventually there should be a logical image pulled from a computer in order to recreate a more professional scenario.

Although the project could start by pulling an image from a Virtual Machine in VMware, it would be more beneficial to create real data on a physical machine. This machine could be a laptop; however, a tablet with a GPS chip in it would be more realistic due to the potential GPS-related artifacts. The tablet will be connected to a Microsoft account, and a Windows Phone should also be connected to this same account. Fake data should be generated via both devices by connecting to various Wi-Fi networks and using maps and social networking apps.

After the data has been generated, the device should be imaged using a write-blocker, FTK Imager, and a Workstation. The extraction may be more difficult on a tablet since the SSD cannot be extracted without destroying the tablet, so alternate extraction methods should be researched.

With the data extracted the analysis can begin, and the artifacts can be compared. Attempts to import into Encase 7, FTK 5.0, or Autopsy can be made, but it is expected that there may be problems since they will not recognize Windows 10.

Equipment Used

1) VMware Workstation 11.0
2) FTK Imager
3) Windows 10 Preview Build 9926 & Build 10049
4) Laptop/tablet capable of running Windows 10
5) Nirsoft Suite

The Software and Hardware setup was the following:

- Single VMWare machine
- One Nokia Lumia 635

VM Hardware

<table>
<thead>
<tr>
<th>VMWare Version</th>
<th>11.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>4 GB</td>
</tr>
<tr>
<td>Processors</td>
<td>1 (Intel Core i7)</td>
</tr>
<tr>
<td>Hard Drive</td>
<td>60 GB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Windows 8.1</td>
</tr>
<tr>
<td>Computer Name</td>
<td>Lcdivm8</td>
</tr>
</tbody>
</table>
VM Hardware

<table>
<thead>
<tr>
<th>VMWare Version</th>
<th>11.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>4 GB</td>
</tr>
<tr>
<td>Processors</td>
<td>1 (Intel Core i7)</td>
</tr>
<tr>
<td>Hard Drive</td>
<td>60 GB</td>
</tr>
<tr>
<td>Operating System</td>
<td>Windows 10 Build 9926</td>
</tr>
<tr>
<td>Computer Name</td>
<td>Lcdivm10</td>
</tr>
<tr>
<td>Time Zone</td>
<td>GMT – 5 (Eastern)</td>
</tr>
<tr>
<td>Username</td>
<td><a href="mailto:lcedivt10@outlook.com">lcedivt10@outlook.com</a></td>
</tr>
</tbody>
</table>

Software Installed

<table>
<thead>
<tr>
<th>Product</th>
<th>Version</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Office</td>
<td>Preview</td>
<td>Regular desktop on Windows 8.1, Touch version on Windows 10</td>
</tr>
<tr>
<td>Facebook</td>
<td>Current</td>
<td>Facebook Modern Application</td>
</tr>
<tr>
<td>Microsoft Solitaire</td>
<td>Current</td>
<td>From the Windows Store</td>
</tr>
<tr>
<td>Tentacles: Enter the Mind</td>
<td>Current</td>
<td></td>
</tr>
<tr>
<td>Modern Skype</td>
<td>Current</td>
<td></td>
</tr>
</tbody>
</table>

Data Collection:
A data generation sheet was involved in creating average data for the user, such as visiting certain websites, installing programs, changing passwords, and deleting and moving files. Data generation created outside of the data generation sheet is documented where appropriate. Almost all data was from Virtual Machines created in VMWare 11.0 running Windows 8.1 fully patched or Windows 10 build 9926. A data generation sheet was created and completed on Windows 10 build 9926 and Windows 8.1. Some artifacts, such as Project Spartan, were analyzed using a separate VM since Spartan did not appear until later in the project.
Analysis

In the beginning, we expected a plethora of differences between Windows 8.1 and 10 due to the large number of changes featured in Windows 10. A lot of the applications in Windows 10 have been re-written, but since they are still based on Windows 8.1, we expected them to only have slight differences.

Results

The following results will be grouped into three categories: updated artifacts, new artifacts, and similar artifacts.

Different/Updated Artifacts

Recycle Bin

One of the most fundamental forensic artifacts in an investigation is the recycle bin. When crimes are committed on computers, one of the first locations to check for evidence is almost always the Recycle Bin. As a result, we will focus on analyzing the recycling bin in Windows 10 as a primary step.

For this analysis we took two nearly identical VMs running Windows 8.1 and Windows 10 and generated data for the recycle bin. Both VMs were logged in to two separate Microsoft accounts, and were running the latest Windows updates as of March 2nd, 2015. Office was also installed on both VMs.

Data Generation

The following data generation tasks were run on March 2, 2015:

<table>
<thead>
<tr>
<th>User Action</th>
<th>Windows 8.1 Timestamp</th>
<th>Windows 10 Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Doc1.docx in Documents folder</td>
<td>19:06</td>
<td>19:08</td>
</tr>
<tr>
<td>Create Pres1.pptx Documents folder</td>
<td>19:08</td>
<td>19:09</td>
</tr>
<tr>
<td>Create Cloud Doc1.docx in Onedrive\Documents</td>
<td>19:11</td>
<td>19:14</td>
</tr>
<tr>
<td>Create Cloud Pres1.pptx in Onedrive\Documents</td>
<td>19:16</td>
<td>19:15</td>
</tr>
<tr>
<td>Create folder “Deleted Folder” in Documents</td>
<td>19:16</td>
<td>19:20</td>
</tr>
<tr>
<td>Create “Folder 1.docx” in Deleted folder</td>
<td>19:17</td>
<td>19:20</td>
</tr>
<tr>
<td>Create “Folder Pres1.pptx” in Deleted folder</td>
<td>19:19</td>
<td>19:21</td>
</tr>
<tr>
<td>Delete Doc1 and Pres1 individually</td>
<td>19:23</td>
<td>19:22</td>
</tr>
<tr>
<td>Delete “Deleted Folder”</td>
<td>19:24</td>
<td>19:24</td>
</tr>
<tr>
<td>Delete “Cloud Doc1.docx” &amp; “Cloud Pres1.docx” in OneDrive\Documents</td>
<td>19:25</td>
<td>19:25</td>
</tr>
</tbody>
</table>
Recycle Bin Analysis
Since Windows 7, Recycle bin artifacts for each user are found in the following location:

DRIVE:\$RECYCLE.BIN\SID

For each file that is deleted, a pair of files is placed in the recycle bin. One file starts with the file name of $I and the other with $R, but both end in the same 6 random characters and the original extension. A screen shot is shown below.

The $I format contains metadata including the file size, deleted time and the file path.

The $R file contains the deleted file itself.

The $I file is formatted in the following manner in Windows 8.1:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length in bytes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8</td>
<td>Begins with 01</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>File Size in bytes</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>Deleted Time (In 64 bit Windows timestamp format)</td>
</tr>
<tr>
<td>24</td>
<td>520</td>
<td>File path</td>
</tr>
</tbody>
</table>

In Windows 10, the contents are still split into these $I and $R files, but the organization of the $I file is slightly different.

Windows 8.1
Below is a screenshot of a $I file in Windows 8.1. As you can see, the offsets match up with the table shown above. The hex is parsed and converted according to the table as well.
Windows 10

Below is a screenshot of a $I$ file in Windows 10. As you can see, the first offset value is 8 bytes long, but it starts with a value of 02. Then, we see the 8 bytes related to the file size, followed by the deleted time matching the data generation sheet. The 4 byte value at offset 24, is the file path length for the deleted file. You take the byte value and convert it to decimal using little endian and add 1 for the trailing null byte.

The rest of the file is no longer 520 bytes and is instead based off the file name, as seen below. It appears that the end of this file is marked by three bytes of continuous zeros.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length in bytes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0*</td>
<td>8</td>
<td>Ends in 02</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>File Size in bytes</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>Deleted Time (In 64 bit Windows timestamp format)</td>
</tr>
<tr>
<td>24*</td>
<td>4</td>
<td>File Path Length</td>
</tr>
<tr>
<td>28*</td>
<td></td>
<td>Dependent upon file path length</td>
</tr>
</tbody>
</table>

* = Changed in Windows 10

Differences

Differences between Windows 8 and 10 are detailed in the table below; minor changes are found in Offsets 0, 24 and 28.
Although the changes are minor, they are significant for tools that rely on the first offset for analyzing the recycling bin. Rifiuti2, for example, will not work with the Windows 10 recycling files, and Encase does not parse the data correctly.

<table>
<thead>
<tr>
<th>Windows 8.1</th>
<th>Windows 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offset 0 begins with 01</td>
<td>Offset 0 begins with 02</td>
</tr>
<tr>
<td>File path is at offset 24</td>
<td>File path begins at offset 28</td>
</tr>
<tr>
<td></td>
<td>Offset 24 is 4 bytes of unknown characters</td>
</tr>
<tr>
<td>File Path is 520 bytes</td>
<td>File path size is dependent upon file path length</td>
</tr>
</tbody>
</table>

**Thumbnails**

Thumbnail artifacts can be important to investigators when dealing with potential evidence found in images. In some versions of Windows, thumbnail data is maintained even when the image itself is deleted. Windows XP had a thumbs.db file that stored the thumbnail image of every file until Windows 7 removed this functionality and replaced the thumbs.db folder with a thumbcache.db file located in:

C:\Users\<USERNAME>\AppData\Local\Microsoft\Windows\Explorer

The thumbs.db file returned in Windows 8 and kept caches of thumbnails in the same folder as Windows XP. Below is a screenshot of the thumbs.db in Windows 8.1 and the absent file in Windows 10.

![Figure 1 Pictures folder in Windows 8.1 (Left) vs. Windows 10 (Right)](image)

Windows 10 removes the thumbs.db file once again, storing the thumbnails in the same location as Windows 7:

C:\Users\<USERNAME>\AppData\Local\Microsoft\Windows\Explorer.

The file header for Windows 10's thumbnail cache is only slightly different from 8.1, and it’s a very simple change. Windows 10 has the value of 0x20 instead of 0x1F at offset 4. When one converts these two hex values to decimal figures, they are 31 and 32 respectively. Below are two screenshots comparing the two file headers of the thumbcache file.
Although a minor change, it affects the tools that help investigators view thumbnail caches, and these tools will need to be updated to become compatible with Windows 10. One item to note is that the thumbnails.db files were present in Windows 10 build 9926, but they are no longer present in build 10049, so it’s possible that this behavior may change before release.

OneDrive
OneDrive log files are stored in two separate but similar locations on Windows 10. The file paths are the following:

<table>
<thead>
<tr>
<th>OneDrive Log Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Windows 8.1</strong></td>
</tr>
<tr>
<td>C:\Users&lt;USERNAME&gt;\Appdata\Local\Microsoft\Windows\OneDrive\logs</td>
</tr>
</tbody>
</table>

Within this folder there are four types of files: SyncEngine.odl, TraceCurrent.ETL, TraceArchive.ETL, and SyncDiagnostics.txt. Below is a screenshot of these files:

The timestamps are not forensically important as the files appear to be created on a routine basis. Generally there are three days’ worth of logs at any given time.
**Functional Differences**

Within these folders, both operating systems have the same file types; the only difference is the way that OneDrive functions on Windows 10. On Windows 8.1, Windows utilizes “smart folders,” a directory that is viewable to the user, but the contents of these directories aren’t actually on the computer. The user has to manually take each directory offline in order to store each file permanently offline otherwise the OneDrive data is only downloaded when needed. What this means for the end user is that he or she could have a computer with only 256 GB of storage and still be able to browse a directory of OneDrive folders that contained 1 TB of data. This also means that logs for OneDrive would reference folders that weren’t physically on the computer.

OneDrive in Windows 10 differs from Windows 8.1 in that “smart folders” are no longer supported. When the user launches OneDrive, they are asked which folders they would like to sync offline, and the only way to view other OneDrive folders is to add more folders. In terms of OneDrive logs, it still looks like folders that aren’t synced all the time are stored in the OneDrive TraceArchive and TraceCurrent .ETL files.

**SyncEngine.odl**

The SyncEngine file in this directory is the most common file found. The .odl file extension is most often used in C++ applications and references many .cpp files such as “filetransferwatcher.cpp” and ”localchanges.cpp.” Each file is created with a timestamp and is exactly 1,025 KB. These files appear to be logs of operations that have been performed, but because of the .cpp file references it’s possible that they are used for the actual function of OneDrive syncing. When a file is synced to OneDrive, a SyncEngine file is created and the file will sync filenames and file hashes among the other logs. It’s possible that OneDrive is submitting these hashes to the OneDrive servers to verify file integrity. Unlike other artifacts below, these logs only contain files that are physically on the computer.

**Trace.ETL**

TraceArchive.ETL and TraceCurrent.ETL are logging files which appear to contain the remnants of the smart folder feature in Windows 8.1. Unfortunately, while event viewer can open them, it doesn’t produce any useful or readable information. However, analyzing the file in notepad seems to work for rudimentary forensics. Below is a screenshot of the contents of the ETL file which references files that are stored only on OneDrive and not physically stored on the device.

![Image of TraceArchive.ETL file](Image)

**SyncDiagnostics.log**

SyncDiagnostics.log is a logging file which displays the operations currently being run on the computer.
actively synced, it will provide a list of pending files. Under normal conditions, it will not have any useful forensic data.

Figure 6 SyncDiagnostics.log file

These files can be opened by event viewer, but they don’t reveal any content in the logs, so they might be modified forms of ETL files. It’s possible that the logging of data that isn’t physically on the drive is a functionality feature leftover from Smart Folders.

Prefetch Files
Prefetch files are used to power Superfetch, Microsoft's system of optimizing program startup speeds and boot times. By analyzing the times in which users open files, Windows can learn the user's behavior and eventually preload the programs before they are launched. Because this information is stored for superfetch, it also makes it ideal for forensic investigators. The location for these prefetch files is located at C:\Windows\Prefetch.

Figure 7 Contents of Prefetch folders in Windows 10

These files store the following information:

- File path of all files accessed by the application during the first 10 seconds of running
- Number of time the application has been run
- The last time the application was run
- Any volumes or folders that were accessed

Windows 8.1 Prefetch files begin with the file signature of 1A 00 00 00 followed by the ASCII characters "SCCA." In Windows 10, the Prefetch file content has little to no resemblance to Windows 8's Prefetch format, and looks to be a compressed MAM file. Unfortunately, this is not compatible with current Prefetch analyzers. Further research will have to be done on this, but it looks like it should be possible to obtain the full uncompressed file if the right tools are written.
New Artifacts

Spartan Browser

Perhaps the biggest change to how forensics investigators will have to analyze Windows 10 is led by the changes to the default browser. Windows 10’s default browser is now a new browser named “Project Spartan.” The code for the browser is based almost entirely on Windows 10’s APIs, which means that the file structure of the application will be somewhat similar to Windows 8 applications.

Artifacts for the latest build of Project Spartan (April 15th, 2015) can be found at the following file path:

C:\Users\<USERNAME>\AppData\Local\Packages\Microsoft.Windows.Spartan_cw5n1h2txyewy\AC\#!001\Spartan

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Type</th>
<th>Date Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppCache</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:22...</td>
</tr>
<tr>
<td>Cache</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:22...</td>
</tr>
<tr>
<td>Cookies</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:55...</td>
</tr>
<tr>
<td>History</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:22...</td>
</tr>
<tr>
<td>IEBingPageDataCache</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:55...</td>
</tr>
<tr>
<td>ICompatCache</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:22...</td>
</tr>
<tr>
<td>ICompatUaCache</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:55...</td>
</tr>
<tr>
<td>User</td>
<td>1</td>
<td>Directory</td>
<td>4/12/2015 7:22...</td>
</tr>
<tr>
<td>$SID</td>
<td>48</td>
<td>NTFS Index All.</td>
<td>4/12/2015 7:22...</td>
</tr>
<tr>
<td>MSIMGSIZ.DAT</td>
<td></td>
<td>Regular File</td>
<td>4/12/2015 7:22...</td>
</tr>
</tbody>
</table>

Within this folder there are 3 obvious artifact locations which will be described below.

Cache

Caches are stored separated in different folders for each page that is cached. The names appear to be random hashed values, and this content is arranged very similarly to Internet Explorer’s cached files.
Figure 10 Spartan Cache in Windows 10 at
C:\Users\<USERNAME>\AppData\Local\Packages\Microsoft.Windows.Spartan_cw5n1h2txyewy\AC\#!001\Spartan\Cache

Below are the cached files of Project Spartan side by side with the cached files of IE. The format looks to be identical.

Cookies
Cookie data is stored in the Cookie folder as expected. The arrangement is almost identical to Internet Explorer’s cookie information.

History
As of today, Microsoft has not added support for browsing the history, and as a result, there isn’t any history data in the History directory, nor is it available to the user. The data is still stored in a folder outside of the packages directory. There is a webcache01.dat file in
C:\Users\<USERNAME>\AppData\Local\Spartan\Database. The .dat file is stored in an ese database format, which is identical to IE11’s webcache.dat file.
Facebook App

One of the most commonly used applications on all mobile platforms is the Facebook application. Released in 2013, the Facebook Windows application runs on all Windows 8.1 and Windows 10 devices. Below is a detailed analysis of the artifacts found in the Modern Facebook application (as of March 16th 2015). The application can be found in the app store here.
**Acquisition:**
The file path for the Facebook databases are located at:

C:\Users\<PROFILE NAME>\AppData\Local\Packages\Facebook.Facebook_8xx8rvfyw5nnt\LocalState\<FACEBOOK_ID>\DB

Within the file structure of this application there are several SQLite databases which include:

- Analytics.sqlite
- FriendRequests.sqlite
- Friends.sqlite
- Messages.sqlite
- Notifications.sqlite
- StickerPacks.sqlite
- Stories.sqlite

These databases can be opened and viewed easily using the sqlitebrowser, which can be downloaded [here](https://sqlitebrowser.org).

**Analytics Database**
The analytics database contains some minor analytical information that Facebook uses to get application feedback. Information provided includes information such as whether or not chat is enabled or what time the user last clicked on the messaging tab.

**FriendRequests Database**
This database contains all pending Facebook requests that the user has with the following attributes:

- Facebook UID
- Time the Friend request was made
- Whether the message has been read
- First and Last name
- Affiliations (Often found to be the school)

**Friends Database**
This database has every single friend stored on the application. In most cases it stores all friends; however, on our test subject with over 600 friends, it only stored 593 of them. This database stores a plethora of information for each friend including the following:

- Facebook UID
- Name
- Contact Email
- Phone Number
- Profile URL
- If the user can receive Push notifications
- If the user has Facebook Messenger
- Communication Rating (Closest friends have higher ranks)
- Birthday Date
Messages Database
This database contains all messages that are still cached on the machine. These messages can be found in the messages table within the database and contain several attributes for each message (listed below). It is unknown how this database chooses which files to download, but some messages listed had timestamps before the applications was installed. For this case, 3,477 messages were stored.

- Thread ID
- Message
- Sender with UID, Name & Email
- Source (Messenger or Web)
- Whether it has been read
- Local Timestamp
- Server Timestamp
- Geolocation Coordinates
- Attachment info

You'll notice that the sender is listed, but not the receiver. To find the receiver, you must go to the threads database and find the Thread ID listed earlier. Then find the receiver's Facebook ID in the "senders column," which lists all recipients of the conversation.

Below is a screenshot that shows a portion of the messages database in SQLite. Notice how the geolocation coordinates are listed despite the device not having a GPS sensor.

Notifications Database
The notifications database handles all of Facebook’s notifications. These are the notifications that pop up when a comment is made, someone "likes" your status, or any other notification occurs. The following useful attributes can be gathered from the notifications table:

- Notification ID
- Object Type (From Facebook stream, Event post, group post, birthday reminder etc.)
- Content of notification ("John Smith likes your status")
- Facebook UID of the person who caused the notification
- Icon URL
- Whether it has been read
- Time in which it was updated
- Time in which the object was created

Stickers Database
This database is where stickers are stored when they are sent or received by the user; this data can be useful
when interpreting messages as the stickers are referenced only by their UID in the Messages database. The stickers table contains the UID of the stickers as well as a link to the sticker in reference.

**Stories Database**

The stories database stores what the user sees on his or her timeline. The stories database is the most useful table within the many database since it contains the content and timestamps of the stories. The following attributes are stored in this table, along with others:

- Story ID
- If the story is Hidden
- Facebook Sub Type (Story, Following, Promoted)
- Attached Story ID
- If the viewer delete the post
- If the viewer can edit the post
- Creation Timestamp
- URL of story:
- Shareable metadata
- Title metadata (includes the text in the story)
- Subtitle
- Icon Image URL
- Edit History

**Similar/Unchanged artifacts**

**Event Logs**

Event logs on Windows 8.1 and 10 can be found in: C:\Windows\System32\WindowsEvt\logs. Below is a screenshot of the event logs when viewed in FTK imager in Windows 8.1 as well as Windows 10. Both are identical formats when placed under a hex editor.
Windows 8 Event logs (Top) & Windows 10 Event Logs (Bottom) (C:\Windows\System32\Windowsevt\logs)

It is possible that some additional services are now logged by Windows 10, and more research will need to be done to find those artifacts.

**Internet Explorer**

Internet explorer data in Windows 10 is still kept in the same location and format as Windows 8.1. Below are screenshots of each artifact. IE history is located at: C:\Users\<USERNAME>\AppData\Local\Microsoft\Windows\WebCache.
Typed URLs that are cached are also still stored in the registry at NTUSER\SOFTWARE\Microsoft\Internet Explorer\TypedURLs and the corresponding time for each URL is in TypedURLs.
USB Activity

USB Activity on Windows 10 is stored in the same registry keys as 8.1. HKLM\System\CurrentControlSet\Enum\USBStor still stores artifacts related to USB devices connected to the computer.
**LNK Files**

Important LNK files, such as recent files and start menu folders, have gone largely untouched. Start menu .lnk files in Windows 10 are stored in the same format as they are in Windows 8.1. Additionally, tools such as LinkParser v1.3 continue to work in the same way. The start menu files can be found at:

C:/ProgramData/Microsoft/Windows/Start Menu/Programs

![Figure 17 Windows 10 Lnk files stored in C:/ProgramData/Microsoft/Windows/Start Menu/Programs](image)

One item to note is that almost all modern applications do not show up in this folder, only desktop applications and the Windows Store.lnk and Immersive Control Panel.lnk shortcuts are displayed here. Windows 8.1 functions similarly as well, so it looks like further research will have to be done to find out what is stored on the start screen.

**Conclusion**

The results of this project show that over half of the core artifacts have changed from Windows 8.1 to Windows 10. The team also discovered artifacts that have never been analyzed before, so it should prompt the community to write and rewrite tools that analyze these artifacts. Unfortunately not all our original goals were met due to more artifacts changing than anticipated, and we hope our work is continued in the future.

The following artifacts have changed since Windows 8.1:

- The Recycle Bin
- Thumbnails
- OneDrive
- Prefetch Files

The following artifacts are new artifacts that are features in Windows 10 or have never been analyzed before:

- Spartan Browser
• Facebook Application

The following artifacts have remained unchanged:
• Event Logs
• Internet Explorer
• USB Activity
• LNK Files

Further Work

Further work needs to be done on the following artifacts:

• Notification Center
• Modern Office
• Synced Data
• Cortana Search History
• Modern Mail Application

The team was able to find relevant artifacts but did not have enough time to document them efficiently. It is highly recommended that the project is continued, and the modern applications are effectively analyzed.

Acknowledgements:

Yogesh Khatri has been a huge help on this project, as well as Kyle Tellers, who authored a Windows 8.1 forensic analysis. The LCDI has created an excellent work environment for such a project, and the team would like to thank everyone who was involved.