Interested in learning more about security?

# SANS Institute InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

## Windows Phone 8 Forensic Artifacts

Due to the fast pace of progression of mobile device technology, a need often arises for forensic examination of mobile devices that are not yet supported for data extraction and parsing by commercially available mobile forensic tools. This is particularly the case with less commercially popular and therefore less supported mobile operating systems including Windows Phone 8. Through real life case study examples and the experiences of law enforcement practitioners, this paper explores the challenges that practitioner...

Copyright SANS Institute

Author Retains Full Rights

AD

# Windows Phone 8 Forensic Artifacts

Cynthia Murphy[a], Adrian Leong[b], Maggie Gaffney[c], Shafik G. Punja[d], JoAnn Gibb[e], Brian McGarry[f]
[a]*Madison Police Department, Wisconsin, USA*
[b]*Independent Digital Forensics Researcher, Sydney Australia*
[c]*Teel Technologies, Connecticut, USA*
[d]*Calgary Police Service, Calgary, Alberta, Canada*
[e] *Ohio Attorney General's Bureau of Criminal Investigation's Cyber Crimes Unit, Youngstown, Ohio, USA*
[f] *An Garda Siochana, Dublin Ireland*

**Abstract**

Due to the fast pace of progression of mobile device technology, a need often arises for forensic examination of mobile devices that are not yet supported for data extraction and parsing by commercially available mobile forensic tools. This is particularly the case with less commercially popular and therefore less supported mobile operating systems including Windows Phone 8. Through real life case study examples and the experiences of law enforcement practitioners, this paper explores the challenges that practitioners have faced with Window Phone 8 devices, the practical solutions to those challenges, and presents from practitioner experience, useful artifact locations from Windows Phone 8 devices.

Keywords: Windows Phone 8; mobile device forensics; digital forensics; mobile forensics tools; Cellebrite; Physical Analyzer; X-Ways; EnCase; Nokia Lumia 520; JTAG; Python scripting

## 1. Introduction

Because of the fast pace of change of mobile device technologies and operating systems, there are times when a newer mobile device which is unsupported or only partially supported by commercial mobile forensic tools for data extraction and parsing must be examined in the course of a criminal investigation, with the end goal being the extraction of digital evidence for use in court. In these cases, novel examination techniques must be developed and used, while still adhering to acceptable digital forensics process [1].

Mobile devices and mobile operating systems that are less popular or less commonly used by the general public are also less commonly seen by forensic examiners in their labs, and are therefore less supported by commercial mobile device forensic tool providers. This includes the Windows Phone 8 operating system, which holds an estimated 2.5% worldwide market share as of August 2014, according to IDC [2], and which fell a percentage point in market share from the previous year.

Less popular devices tend to garner less development attention from commercial mobile forensic tool companies, as they tend to develop their tools for the broadest coverage of devices most commonly seen and examined by their customer base. While companies such as Cellebrite are receptive to assisting law enforcement with specific development needs for unsupported or less supported phones in cases where the need is compelling, often the timetable of an investigation doesn't align with the software development priorities and focus of the commercial mobile forensic tool providers.

Another compelling reason to develop alternative forensic methods to the available commercial tools is validation. Providing non-commercial alternatives and methodologies to address data from mobile devices allows results to be compared to those provided by the commercial tools to determine the efficacy of the commercial tools, and to ensure that they are fully addressing the available data.

When commercially available tools don't support or only partially support data extraction and parsing from a phone, then the forensic examiners recourse is to resort to manual methods to document data in the phone. This process entails that a forensic examiner take digital photographs or video of the display of the device to record data that is of probative, evidentiary and/or investigative value. The primary benefit to this method is that anyone reviewing the photographed content will observe the data from the perspective of the device and the device user.

However there are a number of drawbacks to the manual method of data documentation. If there is an extensive amount of data to photograph this method is time consuming and inefficient. If the device screen or navigation controls are damaged, without physical repair of the device, taking photographs will not work. Also, the various major smart phone micro operating systems do not always use the same device

navigation methods, and a forensic examiner can inadvertently cause artifacts to be created (such as an accidental outgoing call) whilst navigating the device. Manually navigating the device can also cause unread flags to be changed to read for artifacts such as email and SMS messages or missed calls. The manual method will not work on PIN or pass code locked devices. Finally, much of the system level data is not available for viewing through the device display and can typically only be obtained from a file system or physical level data extraction.

As a result of these challenges, informal networks of forensic examiners from law enforcement, academia, and the private sector who are working on similar mobile forensic challenges arise, and those individuals work together to move solutions forward. These networks may also include developers from commercial mobile forensic tool companies working in parallel with examiners on a case by case basis.

## 2. Case Example: Madison, Wisconsin Home Invasion \ Sexual Assault

In February of 2013, a home invasion and sexual assault occurred in the City of Madison, Wisconsin. The home invasion involved a conspiracy, planned by a drug addicted escort who solicited the assistance of her drug dealer in order to rob the man she was temporarily staying with, who kept significant amounts of cash in his home from his tattoo business. Unfortunately, the five people carrying out the armed robbery plan forced entry into the wrong half of the targeted duplex. During the course of the home invasion, a woman who was six months pregnant was sexually assaulted repeatedly in front of her husband by several of the assailants [3, 4].

A Nokia Lumia 520 cell phone running Windows Phone 8 was used extensively during the planning of the robbery. The escort and her drug dealer used the phone to communicate via Facebook messages, SMS text messages, and phone calls, and to communicate with co-conspirators. A text message containing an image of a handgun used in the robbery was sent just hours before the robbery. During the course of the crime, hearing the commotion next door, the prostitute who originally helped to set up the robbery used Facebook Messenger to send a message to the assailants that they had hit the wrong house. Additionally, the phone was used after the crime to produce a video which involved the gun used in the home invasion and assault [3, 4]. In short, the evidence contained on the Windows Phone 8 device was crucial to the investigation and prosecution of the case.

The Nokia Lumia 520 was submitted for forensic examination, pursuant to a warrant based upon the above information. Other than extraction of user-created media files stored on an installed Micro SD card, the Windows Phone 8 based device was not supported by any commercial mobile forensic tool for data extraction. In the Madison case, there was no Micro SD card installed in the phone.

An additional investigative and forensic challenge was that the Windows 8 Phone was discovered to be screen locked, and protected by a four digit pass code. Review of information provided by Microsoft related to Windows Phone 8 suggested that pass code protection of the phone would potentially result in encryption of the data on the device. According to Microsoft's whitepaper entitled Windows Phone 8 Security Overview, "Windows Phone 8 uses BitLocker technology to support the encryption of all internal data storage on the phone with AES 128. After BitLocker is enabled, the phone automatically begins encrypting the internal storage. The encryption key is protected by the Trust Platform Module (TPM), which is bound to UEFI Trusted Boot to ensure that the encryption key will only be released to trusted boot components. With both PIN-lock and BitLocker enabled, the combination of data encryption and device lock would make it extremely difficult for an attacker to recover sensitive information from a device [5]."

Given the compelling nature of the investigation in this case, there was little choice but to attempt to find alternatives to commercial mobile forensic tools for data extraction and parsing of data from the Nokia 520 Windows Phone 8 device. After consultation with other mobile forensics professionals, criminal investigators, and the district attorney's office, an additional search warrant was obtained to authorize extraction of data from the device using the JTAG method, with the recognition that the resulting data might potentially be encrypted and unusable.

## 3. Joint Test Action Group (JTAG) Extraction of Windows Phone 8 from Nokia Lumia 520

The JTAG method is an acquisition procedure that involves connecting to the Standard Test Access Port (TAPs) on a cell phone and then using specialized software to instruct the processor in the cell phone to transfer all of the raw data stored on memory chip(s) within the device to a raw binary file. JTAG is an acronym for Joint Test Access Group, a standard within the Boundary Scan Protocols that was

established to ensure the quality and functionality of the electronics on printed circuit boards or PCBs.

JTAG is a non-destructive process used to bypass security measures and access the memory of the Nokia mobile phone. It is an effective technique to extract a full physical image from devices that cannot be acquired by other means. JTAG results in a raw data dump, and additional analysis work is required to interpret the extracted data. The Windows Phone 8 operating system is relatively new and not in common use when compared to Android or iOS devices. Therefore the data structure and parsing of user data is not currently supported well by most commercial mobile forensic tools.

The Madison Police Department and the State of Wisconsin do not currently have the equipment available to perform JTAG extractions, so a private company was engaged to perform JTAG extraction of the data from the device. During the JTAG process, the phone was disassembled down to the board, which revealed the JTAG ports. Wires were soldered to specific ports on the board. These wires were connected through a bridge to a RIFF Box, one of the many JTAG boxes available. An ATF Box can also be used successfully for JTAG extraction of data from Windows Mobile 8 devices and has an adapter for the Nokia 520 which negates the need for soldering during the JTAG process.

RIFF software was configured for settings consistent with the specific model of Nokia phone. A read command was executed and the data was acquired from the memory of the Nokia Lumia 520.2 RM-915 in the form of a binary file. The read process lasted several hours and upon completion of the read process, the binary file was saved. The wire connections were dismantled and the Nokia was de-soldered and then reassembled to working order and returned to the Madison Police Department along with the resulting 7.25 gigabyte .bin file containing the full physical dump of the data from the memory of the Nokia Lumia 520 mobile phone. Forensic examination was then completed by the Madison Police Department digital forensics unit.

Due to a request for speedy trial, critical time constraints in the investigation and prosecution of the case resulted. Authorization was obtained from the district attorney's office to obtain outside assistance from Cellebrite and others for decoding and parsing portions of the data in this case. Cellebrite developers worked separately but concurrently on decoding SMS Messages, Contacts, and call history. Their resulting coding efforts will be incorporated into an upcoming release of Physical Analyzer.

## 4. Forensic Examination of JTAG Extraction from Nokia Lumia 520

Several commercial mobile device forensic tools were used to try to open the extraction and read the file system without success. Traditional forensic tools were then utilized, and the .bin file was successfully recognized by X-Ways ver. 17, which automatically parsed 28 individual partitions within the data dump, including a number of FAT12 and FAT16 partitions and two NTFS partitions. EnCase 7 will also automatically parse the Windows Phone 8 file system.

No single tool was able to provide adequate coverage of the forensic artifacts from the Windows Phone 8 device, which included a number of Windows based artifacts as well as artifacts and data structures more common to mobile device forensics. EnCase 6.19, EnCase 7, and X-Ways 17 were used along with Cellebrite Physical Analyzer, IEF 5.8, Oxygen Forensic Suite, Epilog, and a variety of open source tools during the forensic examination of the device.

The two NTFS partitions included one that contained system data (Partition 27) and one that contained user data (Partition 28). Partition 27 and Partition 28 included some familiar Windows folder structure as shown expanded in Figure 1 and Figure 2.

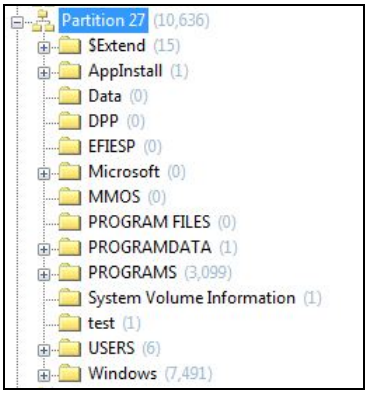



Figure 1- Windows Phone 8 System Data: Partition 27 Shown in X-Ways

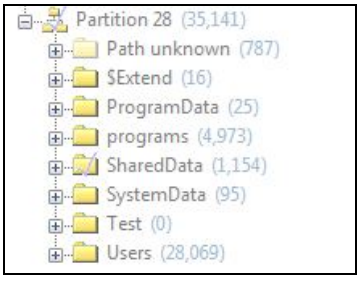



Figure 2 - Windows Phone 8 User Data: Partition 28 Shown in X-Ways

Concurrent examination of the data was conducted using using X-Ways and EnCase, and individual files that were found to contain information of significant evidentiary value were exported for additional analysis in UFED Physical Analyzer 3, and other tools.

## 5. Significant Artifact Locations from the Windows Phone 8 File System

User data from Partition 28 that was found to be of significant evidentiary value was located in the following areas on the Nokia Lumia 520 Windows Phone 8 file system:

### *5.1 SMS Messages and Contact Data:*

Data and content associated with SMS messages and contact information were found within a file named store.vol. The store.vol file was located at Users\WPCOMMSERVICES\APPDATA\Local\Unistore\ store.vol. Specific information regarding the data structure of the store.vol file will be covered later in this paper.

### *5.2 Call History Data:*

Call history data was found within a file with no file extension named phone. The phone file was located at Users\WPCOMMSERVICES\APPDATA\Local\ UserData\phone. A consistent GUID value of B1776703-738E-437D-B891-44555CEB6669 was noted to be present at the end of each call record.

### *5.3 Internet History and Cookies:*

Internet browsing history and cookies were located at Users\DefApps\APPDATA\INERNETEXPLORER\INetCache\. Additionally, default bookmarks were located at SharedData\InternetExplorer\Favorites. Cached Internet History was found at Users\DefApps\APPDATA\Local\Microsoft\Windows\WebCacheV01.dat.

### *5.4 User created Pictures and videos:*

Pictures and videos that had been taken with the embedded camera in the phone were located at Users\Public\Pictures\CameraRoll\. The naming convention for images and videos taken with the Windows 8 phone were formatted as WP_YYYYMMDD_###.jpg. Each image and video had the letters WP (likely for Windows Phone) followed by an underscore, then the four digit year, two digit month, two digit day, another underscore, and then a sequential number, as shown associated with the image below which has the file name WP_20140223_002.jpg:

Figure 3 - Example Image from Nokia Lumia 520 Windows Phone 8: File Name “WP_20140223_002.jpg”

Images taken with the phone contained all of the expected metadata generally found within an image created with a smartphone including date and time, camera settings, and the make and model of the phone used to take the picture.

User created images and videos can also be found on Micro SD cards inserted in Windows Phone 8 devices for data expansion. Micro SD cards installed in Windows Phone 8 devices can only be used to store media files, and cannot be used to store data generated through use of third party apps. Also, although the Windows Phone 8 operating system and user data partitions may be encrypted, files that are stored on SD cards in the phone are not encrypted [5].

### *5.5 Multimedia Messages (MMS):*

Multimedia Message (MMS) Attachments, Formatting Information, and Message Content were found stored on the phone at SharedData\Comms\Unistore\Data in various subfolders as shown below.

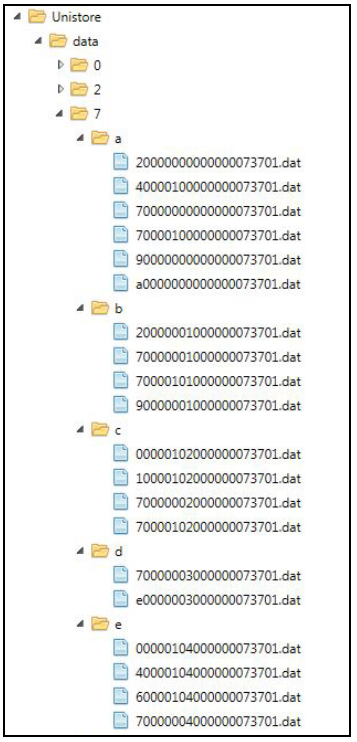



Figure 4- MMS Related data in SharedData\Comms\ Unistore\Data (Cellebrite Physical Analyzer)

The .dat files within the individual folders within the folder structure at SharedData\Comms\ Unistore\Data are not organized by message. In other words the .dat files within a given subfolder named "a" or "b" etc… do not originate from the same MMS message. Rather, the content of an individual MMS message, including text, media file content (image, audio, video), and formatting related data, may be spread across multiple folders at this location.

The associated parts of each MMS message do appear to have similar file names; however every .dat file with a similar file name is not necessarily associated with the same MMS message. For example, the .dat files associated with the gun image in Figure 4 being sent from the phone in an MMS message are named 6000010**a**000000073701.dat (containing the gun picture), 6000010**b**000000073701.dat (containing the textual content of the message), and 6000010**c**000000073701.dat (containing formatting information) shown below in Figures 5, 6 and 7:

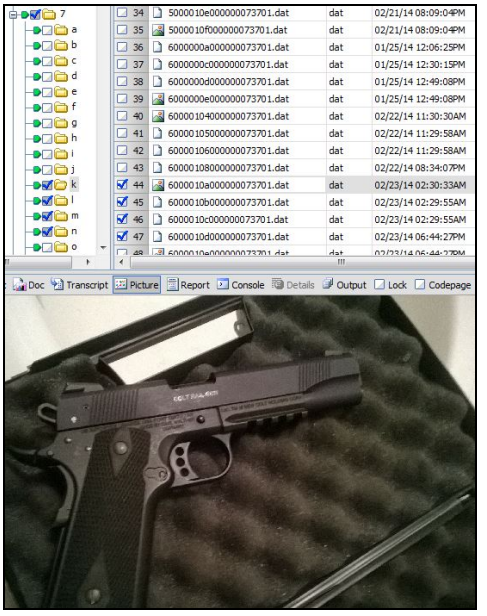



Figure 5 - MMS Message Parts (Image) – Shown in EnCase

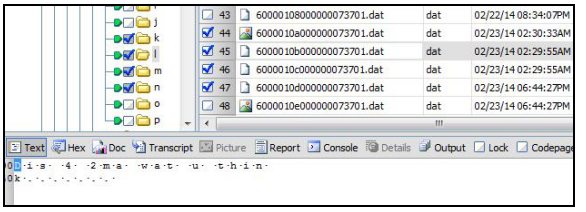



Figure 6- MMS Message Parts (text message content)

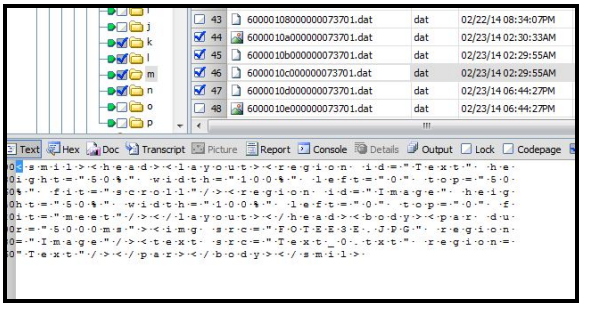



Figure 7 - MMS Message Parts (formatting information)

While the three parts of the message are associated with each other, they are stored in separate subfolders (k, l, and m) within SharedData\Comms\ Unistore\Data. Also note that there is a similarly named .dat file (6000010d000000073701.dat) in the subfolder named n which is actually associated with a completely different message sent later in the day. For this reason, sorting both by file name and then by

created date can be an effective method for grouping content from associated MMS messages together correctly.

As shown in Figure 7, the original file name for the attached file is reassigned, in this case from WP_20140223_002.jpg to FOTEE3E.jpg. A search across the data from Partition 28 can provide further information about the actions performed on that file by the Windows Phone 8 OS in the process, and includes hits in pagefile.sys, the store.vol file, the .dat file from the MMS Message, and USS.log file:

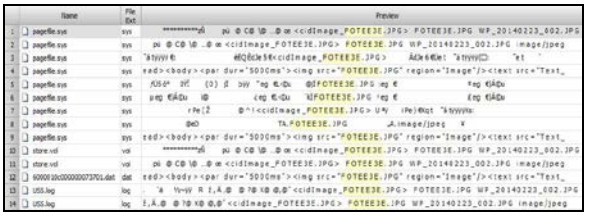


Figure 8 - Keyword search for file name across Windows 8 Phone data - EnCase

Content of file attachments from incoming and outgoing MMS Messages was also located at SharedData\ Comms\Messaging\Temp\MMS. This included some images that were identical to cached image content from multimedia message attachments found at Users\ WPCOMMSERVICES\APPDATA\ Temp\RequestManager\Cache.

*5.6 Pictures Saved from Other Sources:*

Pictures that were saved to the memory of the device which originated from other sources (in this case, images saved from Facebook postings) were found at Users\Public\Pictures\SavedPictures\.

*5.7 OneNote Application Content:*

User content from the OneNote application installed on the phone was located at Users\DefApps\APPDATA\OFFICE\Temp\OneNote. There was additional cached user content from the OneNote application located at Users\DefApps\ APPDATA\OFFICE\Temp\OneNote\OneNoteRunti meCache\OneNoteRuntimeCache_Files. Additional information related to OneNote was also located in the store.vol file, but has not been fully explored as of this time.

*5.8 Artifacts of User Text Entry:*

Artifacts related to user entered text and form history were located on the device at SharedData\Input\neutral\ within files named **ihds.dat** and **livehds.dat.** The content of the ihds.dat file appeared to be a list of unique words input by the user. User entered text consistent with filling out internet forms was found in livehds.dat.

*5.9 User Pass Code:*

The user's four digit PIN code in this case was found in an incoming text message which stated "Da code 0103" as the result of keyword searches for pass code related terms. Understanding that the "Luck of the Irish" might not be with every investigator, a keyword search was conducted for the now known PIN number across the data from the phone, resulting only in erroneous hits. It does not appear that the PIN code is stored in plain text within the data, and it is likely it is stored as a hash instead. Registry entries were found related to PIN code configuration in the Software registry hive at **Microsoft\Comms\ Security\Policy\LASSD\LAP\lap_pw**, however the value itself has not yet been located.

*5.10 ESE Database & Rollback Files:*

According to Microsoft [6], ESE (Extensible Database Files, .edb files) is an advanced indexed and sequential access method (ISAM) storage technology. ESE enables applications to store and retrieve data from tables using indexed or sequential cursor navigation and includes a crash recovery mechanism so that data consistency is maintained in the event of a system crash. ESE also allows for backup and restoration of stored data. The store.vol and phone files mentioned earlier are examples of ESE files found in Windows Phone 8 devices.

The **USS.log** file was mentioned in the previous section and is an ESE database transaction rollback file. During forensic examinations by the team, each Windows Phone 8 device examined was found to contain one or more **USS.log** and **USS#####.log** files (where # represents a numerical digit) which contained extensive SMS and MMS message content, found to contain data from both active and deleted messages. Data was found to be stored using various encoding schemas including UTF-16 and hexadecimal code.

Additional SMS content was found in **USStmp.log,** and the messages found there predated message content from the other recovered USS log file messages, indicating that the USStmp.log file is potentially a copy of a previously existing USS#####.log file.

*5.11 Windows Phone 8 Registry & Other Files of*

*Potential Interest:*

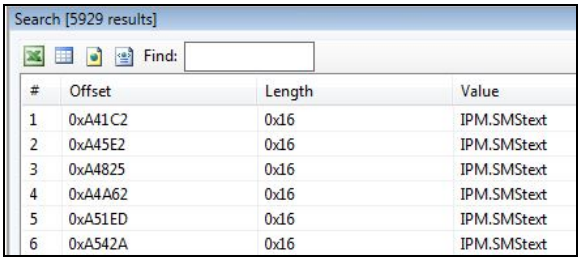
Windows Phone 8 was found to contain versions of the expected registry hives commonly seen in other Windows operating systems, including NTUSER.dat, SAM, SECURITY, SOFTWARE, SYSTEM, and DEFAULT hives. Evidentiary data may also be located in other various Windows artifact locations which are also common to PCs, including, pagefile, $LogFile, $MFT, and $USNJrnl. Sample data was provided to the developers of RegRipper and TriForce for development purposes. Notably, there were no hibernation files present in the data from the Windows Phone 8 devices examined by team members.

## 6. Python Scripting to Parse data from Store.vol and Phone Files:

The volume of potential SMS, MMS and Call History records suggested an automated (scripted) solution. Python is portable/multi-platform scripting language that is widely used in digital forensics and had also been used for previous scripting projects by the team. Based upon the facts of the case, the automated retrieval of SMS content from store.vol was deemed the initial priority.

The first step was finding and understanding the data structures used by Windows Phone 8 to store SMS data. The store.vol file was inspected using a hex editor. Within store.vol, there were numerous textual sentences consistent with text messages, each of which appeared after an UTF16-LE encoded string "IPM.SMStext".

One of the suspected SMS messages contained the text “Da code 0103” which led to the investigatory supposition that this might be the pass code for the phone. After entering the code into the phone, the phone was successfully unlocked, and could be used for comparison purposes for decoding and scripting. The consistent presence of the string "IPM.SMStext" before text message content provided a keyword to search for through the raw data in order to determine the SMS message count as shown in Figure 9.


Search [5929 results]

Find:

| # | Offset | Length | Value |
|---|---|---|---|
| 1 | 0xA41C2 | 0x16 | IPM.SMStext |
| 2 | 0xA45E2 | 0x16 | IPM.SMStext |
| 3 | 0xA4825 | 0x16 | IPM.SMStext |
| 4 | 0xA4A62 | 0x16 | IPM.SMStext |
| 5 | 0xA51ED | 0x16 | IPM.SMStext |
| 6 | 0xA542A | 0x16 | IPM.SMStext |

Figure 91 - Keyword search across raw data to identify SMS Message count - Physical Analyzer 3.

The next step was to locate the encoded timestamp data for each "IPM.SMStext" record. Using the hex interpreter in Physical Analyzer, four potential 8 byte MS FILETIME timestamps were identified. Complicating matters, not all "IPM.SMStext" records had phone number fields and there appeared to be multiple MS FILETIME fields associated with each "IPM.SMStext" record.

By comparison of the data back to the original phone, it was determined that the "IPM.SMStext" records without phone numbers corresponded to Sent SMS messages, while those with phone numbers included were associated with received text messages.

The multiple MS FILETIME fields were sometimes set to the same value but other times varied by seconds or even minutes. By performing a keyword search for each of the MS FILETIME values from selected "IPM.SMStext" records, it was determined that only the second FILETIME field (FILETIME2) consistently occurred twice in store.vol; once in the "IPM.SMSText" record and again in a newly discovered "SMS" record in a different area of the store.vol file. These "SMS" records also contained the destination phone number for Sent SMS which was confirmed via inspection of the phone [6].

A Python script was written to extract relevant SMS data once it was known where and how to identify sent and received SMS message entries. The script is designed to search for "SMS" strings and to store the time and phone number information for each "SMS" record located. It next searches for "IPM.SMStext" strings and extracts the FILETIME2, the sent/received text and associated phone numbers, if available. If the phone numbers are not available in the "IPM.SMStext" record, as is the case for sent SMS messages, the script uses the FILETIME2 value to find the corresponding "SMS" record's phone field. For ease of display and documentation, the script outputs this data sorted by FILETIME2 in Tabbed Separated Variable (TSV) format.

When the script was tested on the data from the Nokia Lumia 520 from the Madison case, the number of script extracted SMS records matched the earlier keyword search count for "IPM.SMStext" performed in Physical Analyzer 3 across the store.vol file. Additional SMS message content can be found by running the described script against pagefile.sys.

Messages recovered from pagefile.sys may be unique from the ones recovered from the store.vol file.

Similar steps were subsequently performed to create scripts for extracting MMS and Contact information from the store.vol file, and Call History from the phone file. The general scripting strategy was to 1) find a record marker field, 2) read and store the surrounding values, 3) sort the extracted records either by timestamp or alphabetically, and 4) output the sorted list to a TSV file.

During the course of the forensic examination, active collaboration occurred with other law enforcement agencies working on similar Windows Phone 8 decoding issues. A second set of "store.vol" data was provided by the Ohio State Attorney General's Office. This allowed for the comparison and detection of commonalities between data sets and ultimately resulted in the production of more flexible and adjustable Python scripts.

It also illuminated discrepancies between the two sets of data. For example, the initial version of the SMS extraction script used a particular byte value at a certain offset within an "IPM.SMStext" record to determine if a message was sent or received. However, this byte flag behavior did not exist in the second set of data (from Ohio), so a new version of the script was written that searched for encoded phone number values in an "IPM.SMStext" record and based the sent/received status accordingly.

It is impractical to attempt to write a script that would consider all possible variances in data structure such as phone number lengths and offsets between fields, so validation is a key requirement of the process. Having retrieved over 5000 SMS messages from the Madison dataset, methods to check the accuracy of the script were needed. Checking each individual SMS was impractical due to the high volume of messages. Random spot checking and manual inspection of SMS messages that were of high evidentiary value was deemed a more time efficient solution.

By manually comparing the output of the script and the data as displayed on the screen of the original evidence (the Nokia Lumia 520 phone) the accuracy of information related to the sender and recipient phone numbers, text message content, and the order in which messages were sent and received could be verified. Unfortunately, the Lumia 520 only displays the date for SMS on the screen, and not the time of the message. However, we could still use the timestamps from various messages to ascertain the order of sent/received SMS and then compare that with the order observed on the phone.

Once the initial versions of the SMS, Call History and Contacts scripts were written, they were then forwarded to other interested parties who then ran the scripts against their own data and provided valuable feedback regarding how well the scripts parsed their data structures.

From this process, it was noticed that although the order of fields remained the same for "IPM.SMStext" records, some datasets from different phones had different offsets between fields. Other data set records, for instance Contacts, had extra field structures compared to the Madison case data. The SMS script was also modified so that it could extract SMS records from the pagefile.sys swap file, after examiners noted additional SMS related data in pagefile.sys.

When comparing notes with other examiners about variation in Call History data sets, a consistent GUID value of B1776703-738E-437D-B891-44555CEB6669 was noted across all available datasets. The value string was consistently present at the end of each Call record which made writing the script significantly easier.

Direct collaboration with the Irish Garda resulted in development of an additional script to parse JSON encoded Facebook Message fields from a file such as pagefile.sys. A set of sanitized representative data was provided to the programmer so that he was able to develop the script without disclosure of confidential case data. This required that someone with access to the sensitive data run the script for testing, and introduced both a time delay and the potential for misunderstanding of script requirements.

**7. Application of scripting to other cases:**

Aside from the main case example discussed in this paper from Madison, WI, collaboration between team members on Windows Phone 8 data structure details and scripting efforts resulted in progress in the following cases:

In June of 2013, the Ohio BCI Cybercrimes Unit in Youngstown, Ohio received an HTC PM23300 mobile phone running Windows Phone 8. The phone had a shattered display and was locked, rendering manual examination of its contents impossible. JTAG extraction of the data using a RIFF Box was completed to create a full physical dump of the raw data stored on the memory chip(s). Due to limited support through commercial cell phone forensic software and knowing the phone utilized a Windows 8 platform, the raw image was loaded into traditional

computer forensic software. The file system was recovered and typical data artifacts were recovered – including graphic images and movies depicting child pornography, some of which appeared to be homemade, as well as web pages and bookmarks. Further analysis revealed partially recovered text messages in the store.vol file, however due to limited software support the messages were not fully decoded and parsed. Through use of the Python script described in this paper, 5,603 text messages were recovered. Examination and scripting efforts continue, as differences in the store.vol files were found across the heterogeneous mobile platforms.

In June of 2014, the Electronic Media Examination Unit in Dublin, Ireland received a Nokia Lumia 520 mobile phone running Windows Phone 8. This phone was believed to contain Facebook messages which were relevant to the case under investigation. The messages were not parsed by commercial mobile phone forensic software and therefore a physical dump of the phone memory was extracted using ATF JTAG. The memory dump was viewed using a file system browser and the Facebook messages in question were found to be stored in pagefile.sys by running a keyword search for some unique message content. The Facebook script was developed based on the Facebook message structure found in pagefile.sys and hundreds of Facebook messages were parsed successfully by the script and precluded investigators having to take hundreds of photographs.

In July of 2014, the Electronic Media Examination Unit in Dublin, Ireland received another Nokia Lumia 520 mobile phone running Windows Phone 8. The phone had been locked out as a result of too many incorrect pin attempts. The only way to gain access to the data was via JTAG extraction. A physical dump of the phone memory was extracted using ATF JTAG. The memory dump was viewed using a file system browser and SMS message content was found in store.vol, USS.log, USStmp.log, USS#####.log and pagefile.sys by running a keyword search for “IPM.SMStext”. Thousands of text messages were parsed from these files using the SMS script described in this paper. The call log and contact scripts were also used to extract call logs from the "phone" file and contact data from store.vol and the USS logs on both phones.

## 8. Conclusions:

Collaboration between forensic examiners across law enforcement, the private sector and academia has resulted in a better understanding of storage mechanisms and forensic artifacts that can be obtained from Windows Phone 8 devices, as well as how to successfully obtain and parse that data.

However, significant work still needs to be done to develop more accessible and automated methods for extracting and presenting the data, and to identify the reasons for the discrepancies in data structures between various devices. Observing further data sets from a wider variety of phone models will help in this regard and is an area for further research. Windows Phone 8 ESE based database files and registry artifacts are other specific areas where more research is required.

## Acknowledgements

Maggie Gaffney, a sworn Law Enforcement Officer who also works for Teel Technologies performed the JTAG extraction of data from the Madison Nokia 520 phone. Her assistance and the ongoing support of Teel Technologies to the mobile forensics and law enforcement communities are greatly appreciated.

After the evidence obtained from the Nokia 520 in the Madison, WI case, and other physical evidence ultimately resulted in all defendants pleading guilty, the Dane County, WI District Attorney’s office authorized use of the data from this phone for the purposes of research on Windows Phone 8 forensics.

Thanks to the input of Brian McGarry and Patrick Morrissey from An Guarda Siochana, additional variations were found and addressed in Windows Phone 8 data structures.

Adrian Leong (aka. Cheeky4n6Monkey) invested countless hours of scripting time and has shared what he learned about Windows Phone 8 with the forensics community through his blog. Gratitude is also expressed to a number of individuals, both managers and developers, from Cellebrite for their contemporaneous scripting and development efforts. Their willingness to work directly with law enforcement officers to provide solutions for unsupported or partially supported phones in ongoing criminal investigations is invaluable.

## References

1. C. Murphy, Mobile Device Evidence Extraction Process V.3, 2013.

2. International Data Corporation. Press release, 8\14\2014. http:\\www.idc.com\getdoc.jsp?containerId=prUS25037214

3. Madison Police Department, Madison, WI. Case # 2014-

56599.

4. http:\\www.channel3000.com\news\police-pregnant-women-sexually-assaulted-in-violent-home-invasion\24904090.

5. Windows Phone 8 Security Overview, Microsoft. October 2013. http:\\www.microsoft.com\en-us\download\details.aspx?id=36173.

6. http://msdn.microsoft.com/en-us/library/gg269259(v=exchg.10).aspx

7. http://cheeky4n6monkey.blogspot.com/2014/06/monkeying-around-with-windows-phone-80.html

**Cynthia Murphy** is a Detective with the City of Madison, Wisconsin Police Department and has been a law enforcement officer since 1985. She is a certified computer forensic examiner and has directly participated in the forensic examination hundreds of digital devices pursuant to criminal investigations of various types of crimes including homicides, missing persons, computer intrusions, sexual assaults, child pornography, financial crimes, and other investigations. She has successfully utilized her skills in the investigation, prosecution, and occasional exoneration in numerous criminal cases involving digital evidence and has testified as an expert in both state and federal court. Det. Murphy earned her MSc. (Hons) from University College, Dublin in Forensic Computing and Cybercrime Investigation, and is a certified instructor and co-author of the SANS 585 Advanced Smartphone Forensics course.

**Adrian Leong** is an Independent Digital Forensics Researcher. He earned his Bachelor of Engineering, Computer Systems (Hons) degree from the University of Technology, Sydney and holds a Graduate Certificate in Forensic Computing from the University of South Australia. He has several years of commercial software development experience and has recently completed various Digital Forensics/Incident Response internships based in the United States. During a two month Visiting Scholar Internship with the Madison Police Department he gained further experience in the extraction of data from mobile devices and assisted with the forensic examination of electronic evidence from mobile devices and computers. Additionally, he was able to use his programming experience to create Python scripts that addressed real life forensic problems. To sharpen his forensic skills and share his research in Digital Forensics, Adrian writes a blog under the Cheeky4n6Monkey pseudonym. This has resulted in various forensic scripts that he has written being shared with the forensics community.

**Shafik G. Punja** is a police officer with the Calgary Police Service and has been a law enforcement office for over 19 years. He has been working in digital forensics since 2003, and has conducted digital forensic examinations on a wide variety of digital data storage devices and operating systems. In 2005, he began researching and developing analytical techniques for mobile devices and smart phone platforms, and has become an expert in the analysis BlackBerry, among other devices. He has also qualified in the Canadian legal system as an expert in the area of digital forensics numerous times, and has had the privilege of being invited as a guest instructor to assist in teaching the Cell Phone Seizure and Analysis Workshop (CSAW) for the Technological Crimes Learning Institute (TCLI) at the Canadian Police College, in Ottawa, Ontario, Canada in 2008 and 2009. Shafik also is a private sector consultant, teaching for Teel Technologies Canada, and is the author of the BlackBerry Forensics training class.

**JoAnn Gibb** is a Computer Forensic Specialist for the Ohio Attorney General's Bureau of Criminal Investigation's Cyber Crimes Unit in the Youngstown Office. JoAnn has been with BCI since 1997. After working for a few years as a programmer, JoAnn joined the Cyber Crimes unit in November of 2003. JoAnn has assisted numerous local, State and Federal agencies involving digital forensics and has testified as an expert witness in several high profile cases. She has a Bachelor's Degree in Information Technology with a Minor in Criminal Justice from Youngstown State University. JoAnn is a Certified Forensic Computer Specialist, a Certified Cell Phone Repair Technician and a Certified Malware Investigator and has over 800 hours of training. JoAnn is an Instructor at Youngstown State University and has given several community seminars on Internet Safety and Cyber Bullying. JoAnn continues her dedication to the mission of the Cyber Crimes Unit of BCI and to the Law Enforcement Agencies of the State of Ohio.

**Brian McGarry** is a Telecommunications Technician with An Garda Siochana based in Dublin. Brian worked as a Test Technician/Engineer with some leading Telecommunications companies for 10 years before joining An Garda Síochána in 2008. He is a certified mobile phone forensic examiner and has been involved in mobile phone forensics since 2013. Brian holds a BSC (Hons) in Information Technology from Dublin City University.

**Maggie Gaffney** recently retired after 21 years with the Massachusetts State Police where she served as a Computer Crimes Investigator and Forensic Examiner. She assists law enforcement, criminal justice agencies, and the private sector with their investigations and cases by conducting JTAG and Chip Off examinations. She is an instructor and mobile forensics expert who teaches advanced mobile data extraction techniques, for Teel Technologies. The courses she teaches enable experienced examiners to learn how to use non-conventional tools like Flasher Boxes, JTAG technology, and the Chip Off process, along with commercial solutions to get the most data from devices.

Last Updated: October 19th, 2017

# Upcoming SANS Training

**Click Here for a full list of all Upcoming SANS Events by Location**

| | | | |
|---|---|---|---|
| SANS San Diego 2017 | San Diego, CAUS | Oct 30, 2017 - Nov 04, 2017 | Live Event |
| SANS Seattle 2017 | Seattle, WAUS | Oct 30, 2017 - Nov 04, 2017 | Live Event |
| SANS Gulf Region 2017 | Dubai, AE | Nov 04, 2017 - Nov 16, 2017 | Live Event |
| SANS Milan November 2017 | Milan, IT | Nov 06, 2017 - Nov 11, 2017 | Live Event |
| SANS Amsterdam 2017 | Amsterdam, NL | Nov 06, 2017 - Nov 11, 2017 | Live Event |
| SANS Miami 2017 | Miami, FLUS | Nov 06, 2017 - Nov 11, 2017 | Live Event |
| SANS Paris November 2017 | Paris, FR | Nov 13, 2017 - Nov 18, 2017 | Live Event |
| Pen Test Hackfest Summit & Training 2017 | Bethesda, MDUS | Nov 13, 2017 - Nov 20, 2017 | Live Event |
| SANS Sydney 2017 | Sydney, AU | Nov 13, 2017 - Nov 25, 2017 | Live Event |
| GridEx IV 2017 | Online, | Nov 15, 2017 - Nov 16, 2017 | Live Event |
| SANS San Francisco Winter 2017 | San Francisco, CAUS | Nov 27, 2017 - Dec 02, 2017 | Live Event |
| SANS London November 2017 | London, GB | Nov 27, 2017 - Dec 02, 2017 | Live Event |
| SIEM & Tactical Analytics Summit & Training | Scottsdale, AZUS | Nov 28, 2017 - Dec 05, 2017 | Live Event |
| SANS Khobar 2017 | Khobar, SA | Dec 02, 2017 - Dec 07, 2017 | Live Event |
| SANS Austin Winter 2017 | Austin, TXUS | Dec 04, 2017 - Dec 09, 2017 | Live Event |
| SANS Munich December 2017 | Munich, DE | Dec 04, 2017 - Dec 09, 2017 | Live Event |
| European Security Awareness Summit & Training 2017 | London, GB | Dec 04, 2017 - Dec 07, 2017 | Live Event |
| SANS Bangalore 2017 | Bangalore, IN | Dec 11, 2017 - Dec 16, 2017 | Live Event |
| SANS Frankfurt 2017 | Frankfurt, DE | Dec 11, 2017 - Dec 16, 2017 | Live Event |
| SANS Cyber Defense Initiative 2017 | Washington, DCUS | Dec 12, 2017 - Dec 19, 2017 | Live Event |
| SANS Security East 2018 | New Orleans, LAUS | Jan 08, 2018 - Jan 13, 2018 | Live Event |
| SANS SEC460: Enterprise Threat Beta | San Diego, CAUS | Jan 08, 2018 - Jan 13, 2018 | Live Event |
| SANS Amsterdam January 2018 | Amsterdam, NL | Jan 15, 2018 - Jan 20, 2018 | Live Event |
| Northern VA Winter - Reston 2018 | Reston, VAUS | Jan 15, 2018 - Jan 20, 2018 | Live Event |
| SEC599: Defeat Advanced Adversaries | San Francisco, CAUS | Jan 15, 2018 - Jan 20, 2018 | Live Event |
| SANS Berlin 2017 | OnlineDE | Oct 23, 2017 - Oct 28, 2017 | Live Event |
| SANS OnDemand | Books & MP3s OnlyUS | Anytime | Self Paced |