12-31-2016

Leveraging the Windows Amcache.hve File in Forensic Investigations

Bhupendra Singh
Defence Institute of Advanced Technology (DU)

Upasna Singh
Defence Institute of Advanced Technology (DU)

Follow this and additional works at: https://commons.erau.edu/jdfsl

Part of the Computer Engineering Commons, Computer Law Commons, Electrical and Computer Engineering Commons, Forensic Science and Technology Commons, and the Information Security Commons

EMBRY-RIDDLE
Aeronautical University.
DAYTONA BEACH, FLORIDA

PURDUE
UNIVERSITY

LEVERAGING THE WINDOWS AMCACHE.HVE FILE IN FORENSIC INVESTIGATIONS

Bhupendra Singh and Upasna Singh

Defence Institute of Advanced Technology (DU)

Girinagar, Pune - 411025

{bhupendra_pcse14, upasnasingh}@diat.ac.in

ABSTRACT

The Amcache.hve is a registry hive file that is created by Microsoft® Windows® to store the information related to execution of programs. This paper highlights the evidential potential of Amcache.hve file and its application in the area of user activity analysis. The study uncovers numerous artifacts retained in Amcache.hve file when a user performs certain actions such as running host-based applications, installation of new applications, or running portable applications from external devices. The results of experiments demonstrate that Amcache.hve file stores intriguing artifacts related to applications such as timestamps of creation and last modification of any application; name, description, publisher name and version of applications; execution file path, SHA-1 hash of executable files etc. These artifacts are found to persist even after the applications have been deleted from the system. Further experiments were conducted to evaluate forensic usefulness of the information stored in Amcache.hve and it was found that Amcache.hve information is propitious to trace the deleted applications, malware programs and applications run from external devices. Finally, comparison of information in Amcache.hve file with information in other similar sources (IconCache.db, SRUDB.dat and Prefetch files) is shown, in order to provide more useful information to forensic investigators.

Keywords: Amcache.hve, Windows Registry Forensics, User Activity Analysis, Program Execution Analysis, Malware Analysis

1. INTRODUCTION

In digital forensic investigations, it is sometimes essential to carry out user activity analysis on a suspected system to know recently accessed file and folders, run applications and their execution history. User activity analysis is also helpful in detection of the anti-forensic behaviors carried out to hinder investigations. The Amcache.hve file is a very useful resource in the area of user activity analysis to analyze recently run applications and their execution history. Earlier, the forensic analysts had access to RecentFileCache.bcf file, which has been replaced by Amcache.hve file in Windows 8 and later versions. Harrell (2013) has highlighted the forensic artifacts stored in RecentFileCache.bcf file. However, the Amcache.hve being newly introduced, is not yet

explored in digital forensics community.

The Amcache.hve stores information related to Windows Application Experience and Compatibility feature in a registry hive file. Forensic analysis of Amcache.hve file can reveal important artifacts such as program name and version, execution file path, file size, installation timestamp, the timestamp of first run, hash of executable file etc. This information persists in Amcache.hve even when the applications and Prefetch folder have been deleted on a user's computer. Therefore, Amcache.hve is also useful in detection of anti-forensic tools, thus, revealing the intentions of a user. Although, the Amcache.hve exposes forensically useful information, it has not been leveraged by forensic analysts as an analytical method for detection of anti-forensic behaviors.

The traces of recently run applications and their usefulness in forensic investigations have been discussed in various literature published. Singh & Singh (2016) have shown the traces of recently accessed files and run applications in Jump Lists on a Windows system. The authors demonstrated that the traces of run applications persist even after the applications are removed from the system. Carvey (2005, 2011) and Wong (2007) have described that numerous artifacts related to run programs on Windows system are recorded in UserAssist registry key. Mee & Jones (2005), Carvey & Altheide (2005) and Mee et al. (2006) have investigated the UserAssist key to identify the external devices that have been connected to the system in order to run a malware program or an anti-forensic tool. Collie (2013) has examined the evidential potential of Windows IconCache.db file and found that Icon-Cache.db file stores artifacts related to executable files when they are present on or run from USB connectable devices. Further, Lee & Lee (2014) identified the structure and highlighted the importance of Icon-

Cache.db file in forensic investigations. The authors demonstrated that IconCache.db file stores full file paths for executable files that have been run, viewed, or installed on a Windows system. Khatri (2015) has examined the SRUDB.dat file in Windows 8 as a new database for tracking user activities and linking launched programs by users. The author has demonstrated the usefulness of SRUDB.dat database in forensic investigations by tracking Windows Store App runs, tracking processes run from external media devices, tracing deleted processes and anti-forensic tools. In recent study, Singh & Singh (2017) provides in-depth understanding of the artifacts and their location created by Cortana application in Windows 10 system. However, we could not find any literature published on forensic analysis of Amcache.hve file except a brief narration by Kim & Lee (2015).

This study mainly investigates the artifacts created and retained in the Amcache.hve file when a user performs certain actions, such as running executable files from host or from USB devices. The effects on Amcache.hve of installing package applications containing many executables files tied into a package, are also considered. The paper provides an overview of how the Amcache.hve file comes into existence and how it grows in size and records the artifacts. The effects of anti-forensic attempts on Amcache.hve are also taken into consideration, however, results of this experiment may vary depending upon the version of Windows operating system. Furthermore, the forensic application of Amcache.hve is evaluated in terms of tracking deleted applications, malware programs and applications run from external storage devices. The paper also identifies the locations and artifacts recorded in interrelated files, however, the prime goal of study is to extract the information retained in Amcache.hve file and its application in

digital forensics.

2. MOTIVATION AND RESEARCH METHODOLOGY

There exists few posts on digital forensic blogs which describe the forensic usefulness of Windows 8 Amcache.hve file. However, the effect of user activities on Amcache.hve file in different distributions of Windows is not yet highlighted in forensic community.

The research method used in this paper is based on observations and experiments. For studying the behavior of Amcache.hve with respect to user activities, several experiments were conducted on three test computers: a laptop running Windows 8 Pro, a desktop running Windows 8.1 Pro, and another laptop running Windows 10 Pro (v1511). A number of user activities were carried out on these test computers in accordance with the objective of the experiment. For extracting Amcache.hve file from test computers, a Live Boot DVD of Ubuntu 14.04 LTS was used. The Amcache.hve file from each test computer was then parsed using Registry Explorer (Zimmerman, 2015). The findings drawn on the basis of observations are discussed in Section 4. Apart from it, Amcache.hve files from several other systems running different distributions of Windows were extracted and analyzed to validate the findings.

3. AMCACHE.HVE FILE IN WINDOWS

Microsoft Windows operating systems use Application Experience and Compatibility feature to ensure the compatibility of software among different distributions of Windows operating system (Microsoft, 2016). For this, Microsoft implements Windows Compatibility Infrastructure (Shim Infrastructure) technique in Windows operating systems. The mechanism of Shim Infrastructure technique is beyond the scope of this research. However, the implementation of this feature generates some interesting artifacts related to program executions that can be leveraged in digital forensic investigations. Davis (2012) leveraged application compatibility cache data to recover artifacts such as file size, file last modified time and last execution time. In this research, we highlight the usefulness of Amcache.hve file in forensic investigations as one of the artifacts associated with Application Experience and Compatibility feature.

The Amcache.hve is a registry hive file introduced in Windows 8 as the replacement of Windows 7 *RecentFileCache.bcf* file under *%SystemDrive% \Windows\AppCompat\Programs* directory. We observed that the Amcache.hve is now also available in Windows 7 systems since October 2015. Depending upon the operating system in use, the Amcache.hve file is stored at following locations:

- **Windows 7/8/8.1:** %SystemDrive%\Windows\AppCompat \Programs\Amcache.hve

- **Windows 10:** %SystemDrive%\Windows\appcompat \Programs\Amcache.hve

The structure of Amcache.hve file follows Windows NT Registry File (REGF) format. Microsoft uses REGF format in Windows NT (or later) to store a part of the Windows registry. There are many software tools (Carvey, 2013; NirSoft, 2013; AccessData, 2014; Zimmerman, 2015) available to parse and view the registry file. The following subsections describe the information recorded in Amcache.hve and its interrelated files.

3.1   The Hive

The Amcache.hve file contains sections of 4KB referred as "bins." As the Amcache.hve file grows, these bins make allocation as well as maintenance of the hive file itself. The Amcache.hve starts with four bytes signature value "regf" (0x72656766). Figure 1 shows the hierarchical structure of Amcache.hve file; seven subkeys (folders) under the 'Root' key were observed. The information of interest to a forensic analyst is located in the 'File' and the 'Programs' subkeys. The subkeys under the File key are the Volume Globally Unique Identifiers (GUIDs) of the mounted devices. The information related to mounted devices can be corroborated with the values of registry key at HKLM\SYSTEM\MountedDevices.

3.2   File References

Each Volume GUID stores file reference keys and each file reference key represents a unique file on disk (Khatri, 2013). The values of the file reference keys were observed from 3 to 23 in Windows 8/8.1/10. Table 1 shows value name, value type and data for file reference keys. Minimum three values were observed under any file reference key namely $15, 17$ and 100. The value 15 stores full path to the executable file from that was run. The value 100 represents the program ID that may be found under the Programs key. The value 101 represents the SHA-1 hash of the executable file preceded by four extra 0s. The last write timestamp on a reference key in Registry Explorer is considered as the timestamp of application's first run. No change was observed in timestamp on subsequent runs of applications. Few file reference keys store all 23 values, thus, containing more information related to that executable file. The value 11 and 17 represent last modified timestamps which were observed to be same when decoded. Major

portion of file reference keys was identified; however, the purpose of few fields (data in values $a, b, d, 10$) remains unknown.

Table 1. Stored information in file reference keys

Value	Value type	Description
0	Unicode	Program name
1	Unicode	Publisher name
2	Unicode	Program version (major)
3	DWORD	Language Code
4	QWORD	SwitchBackContext
5	Unicode	Program version (minor)
6	DWORD	File size (in bytes)
7	DWORD	PE Header field-SizeOfImage
8	Unicode	Hash of PE Header
9	DWORD	PE Header field-Checksum
10	DWORD	Unknown (always 0, 6 or 13 in all tests)
11	FILETIME	Last modified timestamp
12	FILETIME	Created timestamp
15	Unicode	Full path to file
16	DWORD	Unknown (0 or 1 for all tests)
17	FILETIME	Last modified timestamp 2
a	QWORD	Unknown
b	QWORD	Unknown
c	Unicode	File description
d	DWORD	Unknown
f	DWORD	Linker timestamp (Unix)
100	Unicode	Program ID
101	Unicode	SHA-1 hash of executable file

3.3   Programs Key

The subkeys under the Programs key are the program IDs assigned to each MSI package. The information stored in a program ID is similar to the information in a file reference key. The number of values in a program ID was observed from 12 to 21 in Windows 8/8.1
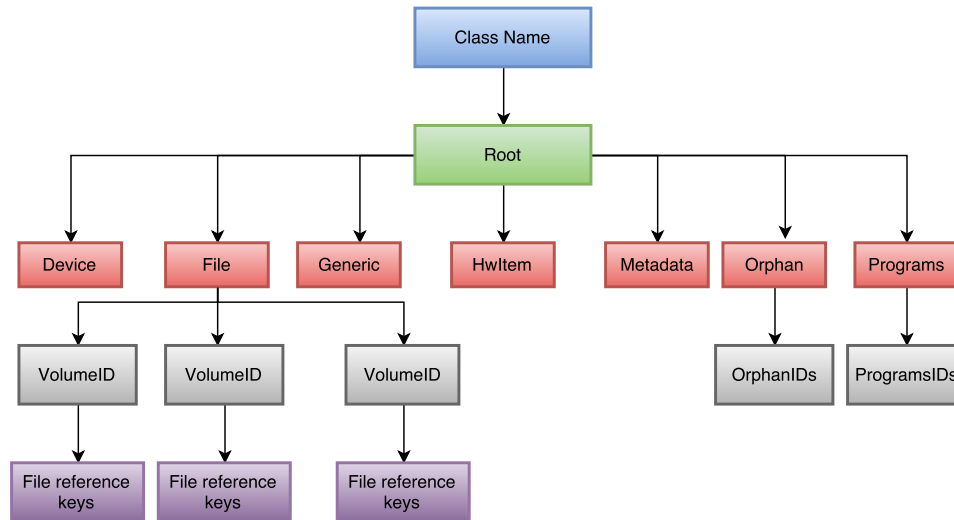
Figure 1. Directory structure of Amcache.hve file

and 16 to 21 in Windows 10 system. Each program ID contains important information related to installed application as summarized in Table 2. For all the cases, the data in value 10 and 12 represents package GUID and was found to be same. Further, the data in value 11 and f describes product GUID and was also observed to be same. The product GUID was also found in uninstall registry key (value 7). The information in value a represents the timestamp of application's installation and the information in value b represents the timestamp when the application was deleted (uninstalled). In the analysis, data in value $Files$ was found as list of files referenced by package applications. Each referenced file under $Files$ list is represented as VolumeGUID@FileRef where FileRef is the file reference key beneath Volume GUID of the File key. In testing, it was observed that not all referenced files in $Files$ list could be found in file reference keys. The relevant fields in a program ID were identified; however, purpose of few fields remains unknown that can be explored in further study.

3.4   Inter-related Files

It was observed that there exists four inventory files associated with Amcache.hve file; however, these files are temporarily stored on disk and automatically created and deleted during user actions. The location of three inventory files: *ApplicationInventory.xml*, *DeviceInventory.xml* and *FileInventory.xml* is under the directory %SystemDrive%\Windows \appcompat\appraiser \Gated in Windows 10. Apart from these three inventory files, one more inventory file associated with Amcache.hve file was observed under the directory %SystemDrive%\Windows \appcompat\Programs. This inventory file consists information related to installed programs, files and Internet Explorer Add-on list.

4.   EXPERIMENTS AND RESULTS

The experiments are divided in three sets: baseline Amcache.hve research (Set A), advanced Amcache.hve research (Set B) and anti-forensic attempts on Amcache.hve (Set C). In Set A, the baseline behavior of Am-

Table 2. Stored information in programs ID keys

Value	Value type	Description
0	Unicode	Program name
1	Unicode	Program version
2	Unicode	Publisher name
3	Unicode	Language Code
5	DWORD	Unknown
6	Unicode	Install source (always Msi, AddRemoveProgram or AddRemoveProgramPerUser in all tests)
7	Unicode	Uninstall Registry key
10	Unicode	Package GUID
11	Unicode	MSI product GUID
12	Unicode	MSI package GUID
13	DWORD	Unknown (always 0 in all tests)
14	DWORD	Unknown (always 0 in all tests)
15	DWORD	Unknown (always 0 in all tests)
16	Binary	Unknown
17	QWORD	Unknown (always 2814749767116800 in all tests)
18	DWORD	Unknown (always 0 or 1 in all tests)
a	QWORD	Install timestamp (Unix)
b	QWORD	Uninstall timestamp (Unix, 0 if program is not uninstalled)
d	Unicode	List of file path
f	Unicode	Product GUID
Files	Unicode	List of file entries in package (VolumeGUID@FileRef)

cache.hve is observed after a clean install of different operating systems. For this, various experiments are performed to observe the initial content of Amcache.hve and interrelated files and content after reboot of operating system. In Set B, various experiments are performed to observe the effects on Amcache.hve file as a result of repeated user activities such as running host-based executables, installing applications from host,

running portable programs from external devices and installing applications from external devices. In Set C, the experiments are carried out to simulate the anti-forensic attempt like evidence destruction (deletion of Amcache.hve file). The objectives of the experiments devised in accordance with these sets are summarized in Table 3. The tools used in this research are listed in Table 4.

The following subsections describe the results of the experiments formulated in Table 3. For performing the experiments in Set A, the following distributions of Windows operating systems were taken into consideration: Windows 7 Professional 64-bit, Windows 8 Pro (Build 9200) 64-bit, Windows 8.1 Pro (Build 9600) 64-bit and Windows 10 Pro v1511 (Build 10240) 64-bit.

4.1 Set A: Baseline Amcache.hve Research

To know the baseline behavior of Amcache.hve file, four Windows systems were installed on different laptops. The results are as follows:

1. The clean install of Windows 7 does not result in creation of Amcache.hve.

2. It was observed that clean installation of Windows 8.1 and Windows 10 result in creation of Amcache.hve file but not always in Windows 8.

3. The number of default applications present in Amcache.hve file depends upon the operating system under consideration and may vary because of the configuration settings selected at the time of installation. Table 5 shows the default applications installed during a clean installation of different Windows operating systems.

4. First reboot of the system results in creation of 'Application

Table 3. Details of experiments devised

Set	Objective	Actions
A: Baseline Amcache.hve research	Establish file provenance	To ascertain: 1. When the Amcache.hve is created. 2. Its content just after creation. 3. Interrelated files and their content. 4. Amcache.hve content following the system reboot or shutdown.
B: Advanced Amcache.hve research	To observe the content of Amcache.hve file and interrelated files after repeated user activities	
B1	To observe the effects of basic user activities	To ascertain: 1. Amcache.hve content after running a host-based executable. 2. Content of interrelated files after running a host-based executable
B2	To observe the effects of installing an application from host	To ascertain: 1. Content of Amcache.hve file after installing an application. 2. Content of interrelated files after installing an application.
B3	To observe the effects of connecting external devices	To ascertain: 1. Content of Amcache.hve file after an executable file is run from a USB drive. 2. Content of Amcache.hve after an executable file is installed from a USB drive.
B4	To observe the effects of Metro Apps	To ascertain: 1. Content of Amcache.hve after installing a Metro App. 2. Content of interrelated files after installing a Metro App.
C: Anti-forensic attempts	To observe the effects of anti-forensic attempts	To ascertain: 1. What happens if the Amcache.hve file is deleted. 2. What happens when interrelated files are deleted. 3. Modifying the timestamps of Amcache.hve.

Table 4. Software tools used in study

Software tool	Purpose
Registry Explorer	To parse Amcache.hve file
DCode[a]	Binary data to date/time converter
HxD Hex Editor[b]	To view/modify raw content of Amcache.hve file

[a] Available from: `http://www.digital-detective.net/digital-forensic-software/free-tools/`
[b] Available from: `https://mh-nexus.de/en/hxd/`

Experience Inventory' file at the same location of Amcache.hve file. The inventory file is named as AEINV_AMI_WER_{MachineID}_DATE_TIME.xml, where DATE and TIME are file creation date and time (in GMT) while MachineID is 16 bytes GUID value generated by Windows system.

5. It was observed that no other associated file(s) was created until no user activity was performed except system shutdown or reboot.

Table 5. Applications in Amcache.hve on initial creation

	Windows 8 Pro (Build 9200)	Windows 8.1 Pro (Build 9600)	Windows 10 Pro (Build 10240)
Size of Amcache.hve (in KB)	256	256	64
Default applications	C:\Windows\System32\csrss.exe C:\Windows\System32\dwm.exe C:\Windows\System32\LogonUI.exe C:\Windows\System32\services.exe C:\Windows\System32\smss.exe C:\Windows\System32\SppExtComObj.Exe C:\Windows\System32\taskhost.exe C:\Windows\System32\taskhostex.exe C:\Windows\System32\wininit.exe C:\Windows\System32\winlogon.exe C:\Windows\System32\oobe \msoobe.exe C:\Windows\System32\wbem \WmiPrvSE.exe C:\Windows\SysWOW64\ByteCodeGenerator.exe	C:\Windows\System32\csrss.exe C:\Windows\System32\dwm.exe C:\Windows\explorer.exe C:\Windows\System32\lsass.exe C:\Program Files\Windows Defender\MsMpEng.exe C:\Program Files\Windows Defender\NisSrv.exe C:\Windows\System32\RuntimeBroker.exe C:\Windows\System32\SearchIndexer.exe C:\Windows\System32\services.exe C:\Windows\System32\SettingSyncHost.exe C:\Windows\System32\smss.exe C:\Windows\System32\spoolsv.exe C:\Windows\System32\svchost.exe C:\Windows\System32\taskhost.exe C:\Windows\System32\taskhostex.exe C:\Windows\System32\ThumbnailExtractionHost.exe C:\Windows\System32\wininit.exe C:\Windows\System32\winlogon.exe C:\Windows\System32\wbem \WMIADAP.exe C:\Windows\System32\wbem \WmiPrvSE.exe	C:\Windows\System32\oobe \setup.exe C:\Windows\System32\csrss.exe C:\Windows\System32\dllhsot.exe C:\Windows\System32\drvinst.exe C:\Windows\System32\rundll32.exe C:\Windows\System32\svchost.exe C:\Windows\System32\winlogon.exe C:\Windows\System32\applicationframehost.exe C:\Windows\System32\dwm.exe C:\Windows\explorer.exe C:\Windows\System32\oobe \firstlogonanim.exe C:\Windows\System32\logonui.exe C:\Windows\System32\oobe \msoobe.exe C:\Windows\System32\utilman.exe C:\Windows\System32\sysprep \sysprep.exe C:\Windows\System32\oobe \windeploy.exe C:\Windows\System32\oobe \oobeldr.exe C:\Windows\System32\wininit.exe C:\Windows\System32\winlogon.exe C:\Windows\System32\vssvc.exe C:\Windows\System32\svchost.exe C:\Windows\System32\sppsvc.exe C:\Windows\System32\spoolsv.exe C:\Windows\System32\services.exe C:\Windows\System32\smss.exe C:\Windows\System32\lsass.exe C:\Windows\System32\csrss.exe C:\Windows\System32\conhost.exe C:\Windows\servicing\trustedinstaller.exe C:\program files\windows defender\msmpeng.exe

## 4.2 Set B: Advanced Amcache.hve Research

Reconstructing the artifacts left by user activities on a computer system is part of many forensic examinations where user activity is of investigators' interest. The following subsections describe the behavior of Amcache.hve and its interrelated files when different user activities are performed on a Windows system. For this, Windows 10 Pro system was taken into consideration and the numerous user activities were carried out including running host-based executable programs, installing of software applications, running an executable file from a USB thumb drive, visiting URLs from web browsers, installing Metro Apps from Windows Store etc. The effects of such user activities on Amcache.hve were observed and the findings are summarized in the next subsections.

4.2.1 B1: Running a host-based executable

It was observed that when a host-based executable is run for the first time after installation of operating system, the timestamp of run along with executable file path are recorded in Amcache.hve. No artifacts related to run count is recorded in Amcache.hve file and its interrelated files. However, the timestamp of last modification is recorded in Amcache.hve file, which is always the timestamp of first time the executable was run. It means that successive runs of the host-based executable does not effect the Amcache.hve file. Figure 2 shows the timestamp of first run and executable file path of notepad application.

4.2.2 B2: Installing a software program from host

For conducting this experiment, Mozilla Firefox was installed from the host and subsequently run for the first time. The Am-
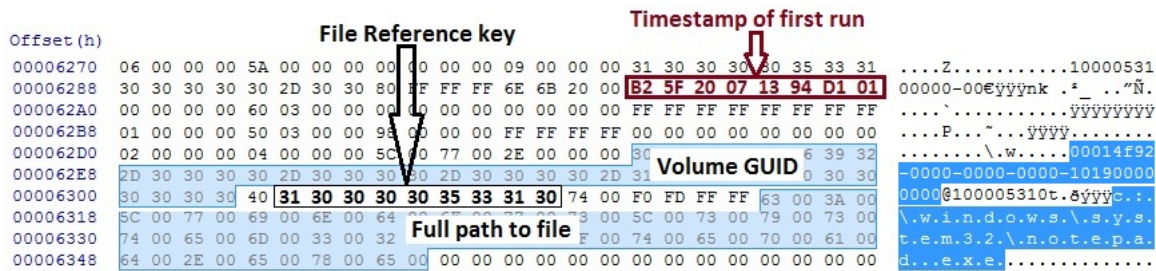
Figure 2. Timestamp of first run and executable file path of notepad application recorded in Amcache.hve

cache.hve file is extracted and then analyzed using HxD hex editor and the artifacts related to Mozilla Firefox were identified. We observed that when a software program is installed then there are two locations in Amcache.hve file at which related artifacts are found. At first location, the full file path from which the executable was run is recorded in file reference key under File key. Figure 3 shows the full file path of executable of Mozilla Firefox from where it was run along with the timestamp of run. At second location in program ID of Mozilla Firefox under the Programs key recorded numerous artifacts such as program name, program version, publisher name, program type etc.

4.2.3 B3: Installing and running a software program from external storage media

This experiment was performed to know the type of artifacts retained in Amcache.hve file when the applications are installed from external devices. A package application (Proxifier Version 3.21 for our case) containing five executables tied into a package was installed from a USB drive to test whether information related to all executables is recorded. Subsequently, the system was allowed to shutdown so that the modifications in the Amcache.hve file could be stored on disk. The Amcache.hve file was then extracted and analyzed using a hex editor. We found program ID, program name and

version, install source, file referenced keys, uninstall registry key and full path to file as shown in Figure 4. The related information such as install timestamp, publisher name, package GUID and volume GUID were also observed in Amcache.hve file which are not shown in the figure for visual convenience. We found five file referenced keys namely: $200014f20$, $200014f13$, $200014f1d$, $200014f22$ and $600014f16$. Each file referenced key corresponds to a referenced file by the package application. For our case, $200014f20$ referred to $Proxifier.exe$, $200014f13$ referred to $ProxyChecker.exe$, $200014f1d$ referred to $SysSettings32.exe$, $200014f22$ referred to $SysSettings64.exe$ and $600014f16$ referred to $unins000.exe$. In addition to this, we also found the created timestamp of each referenced file with their file path in the Amcache.hve file.

The information related to referenced files by a package application can be very useful to forensic/malware analysts. The analysis of Amcache.hve file can provide the name of all referenced files by a malicious program.

When a portable program is run from USB drive, the artifacts of such portable programs are recorded in Amcache.hve file. Figure 5 shows various artifacts related to DCode program which was run from a USB drive, including timestamp of run, program name etc. It was observed that no information related to USB Volume label is recorded in Amcache.hve file; however, Amcache.hve file

Figure 3. Timestamp of run and installed file path of Mozilla Firefox in Amcache.hve

Figure 4. Recorded information related to application installed from USB drive

stores the drive letter assigned to a USB drive. Apart from it, Amcache.hve also stores Volume GUID of USB and the full path of executable file placed on USB drive.

Further, no information related to previous runs of DCode was found in Amcache.hve file. Also, we could not find any information related to applications when the applications

are run from network drives.

4.2.4   B4: Installing a Metro App from Windows Store

This experiment was performed to investigate the effects of installing a Metro App from Windows Store on Amcache.hve. For this, a laptop running Windows 10 Pro was considered and Facebook App was installed from Windows Store. Subsequently, Amcache.hve file was extracted and parsed to observe the artifacts related to Facebook App. It was found that no artifacts related to Metro Apps is recorded in the Amcache.hve file; however, information related to App under consideration were found in *ApplicationInventory.xml* file. Figure 6 shows information related to Facebook App contained in ApplicationInventory.xml file such as program name, publisher name, installation timestamp etc. As discussed in Section 3, these inventory files are not permanently stored on disk volume. Also, the traces of evidence of Metro Apps were not always found in this inventory file.

4.3   Set C: Performing Anti-Forensic Attempts on Amcache.hve

This experiment was conducted to simulate anti-forensic attempts carried out on Amcache.hve file to mislead a forensic investigator. For this, Amcache.hve and associated inventory files were deleted from a system running Windows 10 Pro. It was observed that users cannot delete Amcache.hve file while Windows is running as this file is open in system; however, deletion can be achieved by mounting Windows system drive from a Live Boot DVD of Ubuntu 14.04 LTS. The findings of this experiment for system under consideration are as follows:

- When the Amcache.hve file was deleted, it was recreated on system reboot with

new created and modified timestamps. Furthermore, it was also observed that if any host-based executable was run or any software program was installed, then new Amcache.hve file was created in the same session with new created and modified timestamps. The finding of this experiment supports the theory that Amcache.hve is always open in system whilst the computer system is running; however, the changes in the Amcache.hve are written to disk when a system reboot, shutdown, sleep, or hibernation state is initiated.

- The new Amcache.hve file was found smaller in size (32 KB) than the original Amcache.hve file (64 KB following a clean install).

- The further activities caused the new information to be stored in the newly created Amcache.hve file updating its modified timestamp. For example, activity performed here on system under review was installing Mozilla Firefox web browser from USB thumb drive. Table 6 summarizes the behavior (created and modified timestamps) of new Amcache.hve file with respect to considered activity.

To check the consistency of the findings listed above, the same experiment was replicated on a Windows 8 Pro 64-bit system and it was observed that the results were different, as discussed:

- When the Amcache.hve file was deleted, it was not recreated on system reboot. However, any user activity followed by system reboot resulted in creation of new Amcache.hve file.

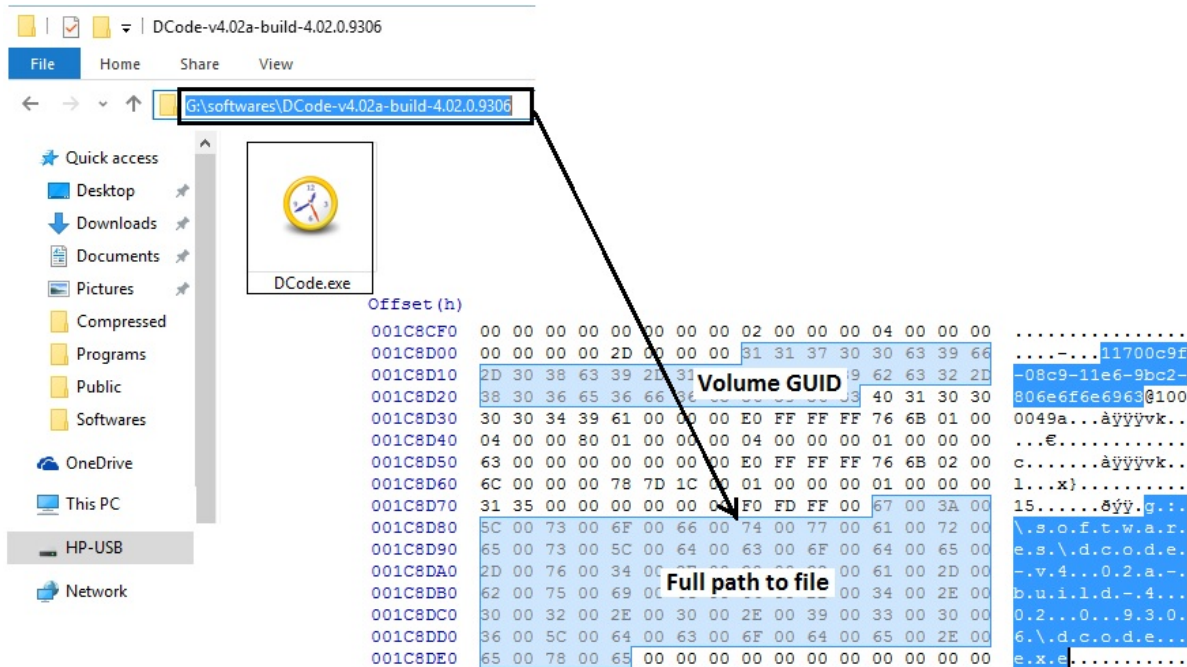- Running a host-based executable or installing an executable produced a new Amcache.hve file with new created and

Figure 5. Recorded information related to portable executable in Amcache.hve

Figure 6. Stored information related to Facebook App in ApplicationInventory.xml file

Table 6. Created and modified timestamps changes to Amcache.hve in Windows 10

Experiment	Activities	Size (KB)	Created timestamp	Modified timestamp
A, 1.1	Clean install Amcache.hve	64	21/04/16 12:07	21/04/16 12:07
A, 1.1	Delete Amcache.hve @12:20, reboot	32	21/04/16 12:20	21/04/16 12:20
B3, 1.2	Install Mozilla Firefox visit URLs, reboot	64	21/04/16 12:20	21/04/16 12:31

modified timestamps. Size of the new Amcache.hve was observed to be same as of the original. Further, the effect of user activities on Amcache.hve file in

Windows 8 Professional was found to be same as in Windows 10 Pro.

4.4 Forensic Utilization of Amcache.hve Information

The Amcache.hve information can be very useful to track the applications run on a suspected system. The following subsections demonstrate the forensic application of Amcache.hve information.

4.4.1 Tracking deleted applications

Previously run applications leave traces in Windows registry, Shortcut files, Jump Lists and in Prefetch folder. However, if the traces from these locations are deleted manually, the information related to run applications can no longer be ascertained; however, the Amcache.hve stores the traces of deleted applications and analyzing it can reveal artifacts of the deleted applications. The analysis can also reveal the timestamp of deletion. Therefore, if the evidence file is deleted, information related to evidence file can be extracted by analyzing the Amcache.hve file.

If a suspected user wipes out all traces of run applications using a privacy protection tool and also cleans the traces of such tools, the investigator cannot find the traces of such anti-forensic behaviors by analyzing the Windows registry, Shortcut files, Jump Lists or Prefetch folder; however, the traces of such privacy protection tools can be obtained by analyzing the Amcache.hve file.

We considered a hypothetical case in which a suspected user wipes out all traces of previously run applications (say, evidence file) by use of a privacy protection tool (CCleaner) followed by the deletion of this tool from the system. The investigator's job is to find out the information related to evidence file and the traces of anti-forensic tool considered in case.

The analysis of Amcache.hve file of the suspected system can reveal the required information to investigator. The investigator can list out all the deleted applications including their execution file path, timestamp created, timestamp of first run, referenced files by applications, timestamp of deletion etc. The Figure 7 shows the traces of CCleaner application in Amcache.hve file when the application has been deleted from the system. The timestamps of applications' installation and deletion can be obtained by decoding Unix 32-bit hex value to date and time format using DCode.

4.4.2 Tracking malwares

The Amcache.hve file records hash (SHA-1) of executable file that can provide immensely useful information to track the applications. Malware analysts can leverage this information to track malwares and their referenced files on a victim system. The traces of run malware persist even if it get deleted or wiped itself from the system.

To test if we can trace a malware, a computer system was infected with a malware sample 'xiaose.exe.' The computer system was restarted and allowed to run malware program followed by shutdown to extract Amcache.hve from Windows system using Live Boot DVD of Ubuntu 14.04 LTS. The extracted Amcache.hve was then parsed using Registry Explorer and traces of malware sample was observed as shown in Figure 8. We found SHA-1 hash of xiaose.exe in value 101 as highlighted in the figure. The hash of the malware was found to be same when it was analyzed on Virus Total where it was detected 54 times as a malicious program out of 56 anti-virus. Thus, the hash of executable files can be very useful information to detect the malicious programs run on a system.

4.4.3 Tracking applications run from external devices

To identify the portable applications run from external devices is a part of many forensic investigations. The analysis of Amcache.hve file can prove to be a use-
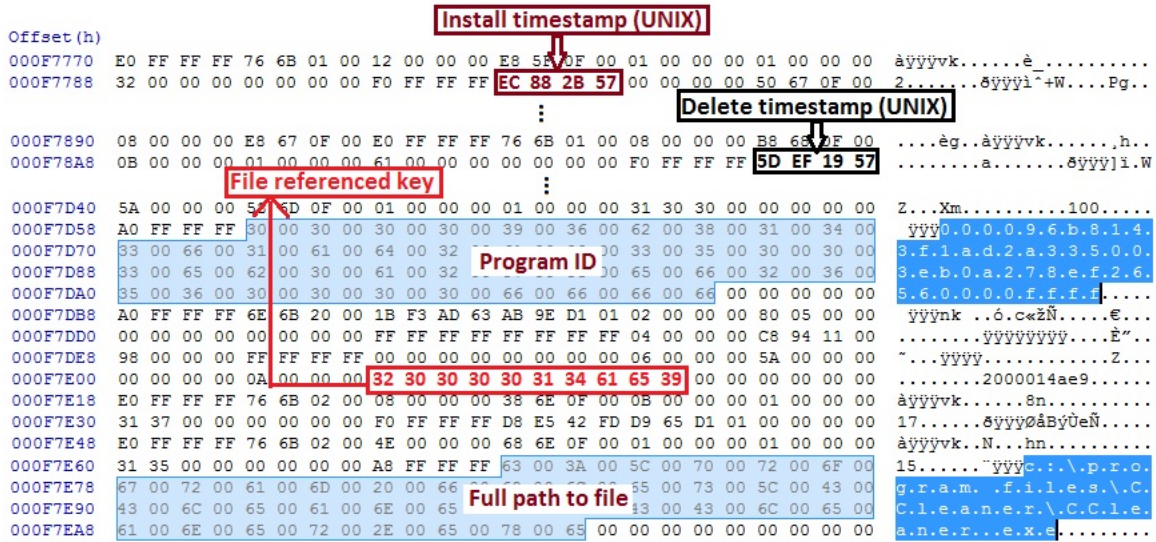
Figure 7. Install and delete timestamps of CCleaner application recorded in Amcache.hve
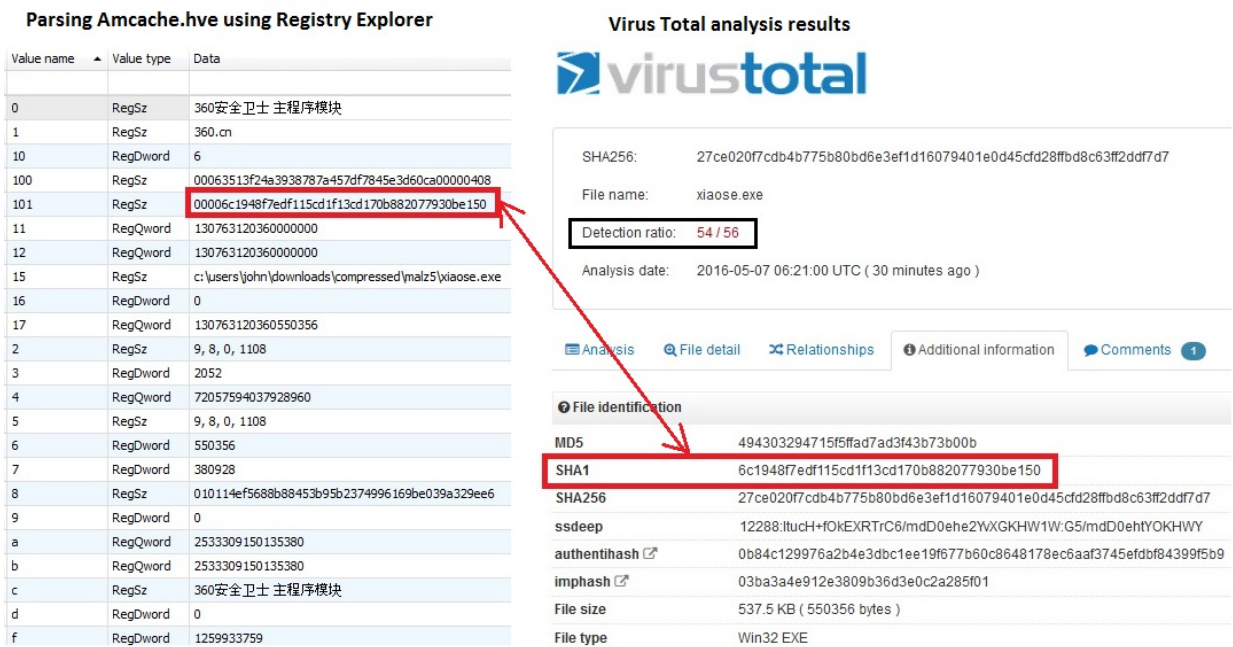

Figure 8. SHA-1 hash of malware program recorded in Amcache.hve

ful forensic utility in the cases where artifacts of external devices mounted on a system have been removed from the *UserAssist* and *MountedDevices* registry key. The Amcache.hve file records the full path to executable file from which the application was executed. The drive letter recorded in this file path can distinguish the fixed and re-

movable devices. In addition to this, the Volume GUID of external device is also recorded in Amcache.hve file. The experimental results discussed in 4.2.3 and shown in Figure 5 that shows the drive letter and the Volume GUID assigned to a USB device from which the portable executable (DCode) was run. However, no information related to drive la-

bel and run count of portable executable was observed in Amcache.hve file.

5. DISCUSSIONS

This section describes the information that can and cannot be gathered from Amcache.hve file and similar sources. We compare forensic potential of similar sources, namely, IconCache.db, SRUDB.dat (present in Windows 8 and higher versions) and Prefetch folder with Amcache.hve file as shown in Table 7. The IconCache.db file records the file paths of applications that have been executed, viewed, stored, installed or copied (Lee & Lee, 2014); however, Icon-Cache.db file does not record any information related to timestamps of the applications. The SRUDB.dat file stores process details including run timestamp, file path and run count of applications; however, the precise startup times are not available from SRUDB.dat file, but it is possible to determine that the application was run on a particular date (Khatri, 2015). Also, the traces from some of deleted applications were not always recorded in SRUM data. Prefetch files can provide information such as application name, file path from which the application was run, created and modified timestamps and run count of applications. However, if an application has been deleted from the system, the delete timestamp is not recorded in the Prefetch file. Also, popular privacy protection tools such as CCleaner, BCWipe, CleanAfterMe or Privacy Eraser are able to delete artifacts in Prefetch files. Fortunately, the Amcache.hve stored artifacts such as the timestamp of an application's first run and traces of deleted applications including the timestamp of deletion, which are not recorded in any other sources considered; however, the information related to run count of applications is not observed in Amcache.hve file. Thus, for better

tracking of applications we can correlate and combine the information in IconCache.db, SRUDB.dat and Prefetch files with the information in Amcache.hve file.

6. CONCLUSION

User activity analysis on a suspected system is part of many forensic investigations. The Amcache.hve file as a new artifact was first introduced in Windows 8 that records the information related to Windows Application Experience and Compatibility feature. Forensic investigators can leverage the information retained in Amcache.hve file to trace the user activities performed on a system specially related to program executions.

This paper investigates the evidential potential of Windows Amcache.hve file and its application in digital forensics. For identifying the artifacts retained in Amcache.hve and investigating the effects of user activities on Amcache.hve, the experiments were devised in three sets. The results of experiments conducted in this research highlight the important advantages of examining the Amcache.hve file which are as follows:

- It is useful to find **recent run applications and their respective history**.

- It represents the information related to **Windows Application Experience and Compatibility feature**.

- The information remains in the Amcache.hve file even when the application and **Prefetch** folder have been deleted.

- Anti-forensic tools such as **CCleaner** are not able to remove Amcache.hve file as it is being used during the Windows runtime;

- Amcache.hve file is an excellent source for **malware analysts to trace program executions**.

Table 7. Comparison of information recorded in IconCache.db, SRUDB.dat, Prefetch folder and Amcache.hve

	Application name and file path	Created timestamp	Last modified timestamps	Run count	Uninstall timestamp	Referenced files	Information related to Metro Apps	Unknown by popular privacy protection tools
IconCache.db	✓	✗	✗	✗	✗	✗	✗	✗
SRUDB.dat	✓	✓	✗	✓	✗	✗	✓	✓
Prefetch	✓	✓	✓	✓	✗	✓	✗	✗
Amcache.hve	✓	✓	✓	✗	✓	✓	✗	✓

It was observed that the Amcache.hve file does not record the run count of applications, so forensic investigators are suggested to combine and correlate the artifacts from other sources such as Windows Shortcut files, UserAssist key and Prefetch folder, for constructing comprehensive activity timelines.

7.   FURTHER RESEARCH SCOPE

The further study on Amcache.hve may be focused on identification of unknown fields in File and Programs keys in Amcache.hve file. The research can also be focused on the analysis of Amcache.hve file when the applications are run, installed or executed from the network drive, Internet or from Windows Store.

ACKNOWLEDGMENTS

REFERENCES

AccessData. (2014). *Registry viewer.* `http://accessdata.com/product-download/digital-forensics/registry-viewer-1-8-0-5`. ([accessed 26-Feb-2016])

Carvey, H. (2005). The windows registry as a forensic resource. *Digital Investigation*, *2*(3), 201–205.

Carvey, H. (2011). *Windows registry forensics: Advanced digital forensic analysis of the windows registry.* Elsevier.

Carvey, H. (2013). *Regripper.* `https://code.google.com/archive/p/regripper/downloads`. ([accessed 26-Feb-2016])

Carvey, H., & Altheide, C. (2005). Tracking usb storage: Analysis of windows artifacts generated by usb storage devices. *Digital Investigation*, *2*(2), 94–100.

Collie, J. (2013). The windows iconcache.db: A resource for forensic artifacts from usb connectable devices. *Digital Investigation*, *9*(3), 200–210.

Davis, A. (2012). *Leveraging the application compatibility cache in forensic investigations.* `http://dl.mandiant.com/EE/library/Whitepaper_ShimCacheParser.pdf`. ([accessed 21-March-2016])

Harrell, C. (2013). *Revealing the recentfilecache.bcf file.* `http://journeyintoir.blogspot.in/2013/12/revealing-recentfilecachebcf-file.html`. ([accessed 14-April-2016])

Khatri, Y. (2013). *Amcache.hve in windows 8 - goldmine for malware hunters.* `http://www.swiftforensics.com/2013/12/amcachehve-in-windows-8-goldmine-for.html`. ([accessed 10-March-2016])

Khatri, Y. (2015). Forensic implications of system resource usage monitor (srum) data in windows 8. *Digital Investigation*, *12*, 53–65.

Kim, M., & Lee, S. (2015). Forensic analysis using amcache.hve. In *Digital forensics and cyber crime: 7th international conference, icdf2c 2015, seoul, south korea, october 6-8, 2015. revised selected papers* (Vol. 157, p. 215).

Lee, C.-Y., & Lee, S. (2014). Structure and application of iconcache.db files for digital forensics. *Digital Investigation*, *11*(2), 102–110.

Mee, V., & Jones, A. (2005). The windows operating system registry–a central repository of evidence. In *Proceedings from e-crime and computer evidence conference* (Vol. 2005).

Mee, V., Tryfonas, T., & Sutherland, I. (2006). The windows registry as a forensic artefact: Illustrating evidence collection for internet usage. *digital investigation*, *3*(3), 166–173.

Microsoft. (2016). *Understanding shims.* https://technet.microsoft.com/en us/library/dd837644%28v=ws.10%29.aspx. ([accessed 09-March-2016])

NirSoft. (2013). *Regscanner.* `http://www.nirsoft.net/utils/regscanner.html`. ([accessed 26-Feb-2016])

Singh, B., & Singh, U. (2016). A forensic
    insight into windows 10 jump lists.
    Digital Investigation, *17*, 1–13.

Singh, B., & Singh, U. (2017). A forensic
    insight into windows 10 cortana search.
    Computers & Security, *66*, 142–154.

Wong, L. W. (2007). Forensic analysis of
    the windows registry. *Forensic Focus*, *1*.

Zimmerman, E. (2015). *Registry
    explorer/recmd version 0.7.1.0.*
    `https://ericzimmerman.github.io/`.
    ([accessed 26-Feb-2016])