

TZWorks® Computer Account Forensic Artifact Extractor (*cafae*) Users Guide



Abstract

cafae is a standalone, command-line tool to extract file and application metadata associated with any Windows user account activity. It can operate on a live target or on separately captured registry hives. All artifacts can be outputted in one of three formats for easy inclusion with other forensics artifacts.

Copyright © TZWorks LLC

www.tzworks.com

Contact Info: info@tzworks.com

Document applies to v0.80 of ***cafae***

Updated: Apr 25, 2025

Table of Contents

1	Introduction	4
2	Location of Hives.....	5
3	How to use <i>cafae</i>	5
3.1	Volume Shadow Copies	7
4	Example User Hive Artifacts that are Extracted and Parsed.....	8
4.1	Metadata Associated with Running a Program	8
4.1.1	Open -> Run Dialog	8
4.1.2	UserAssist Key	9
4.1.3	ProgramsCache Key	11
4.1.4	MUICache Key	12
4.1.5	Run Key and Miscellaneous Applications.....	13
4.1.6	FeatureUsage	14
4.2	Metadata Associated with Viewing/Opening/Editing Files	15
4.2.1	Recent Documents and Associated Keys	15
4.2.2	JumplistData Key	17
4.2.3	StreamMRU Key	17
4.2.4	Open -> Save Dialog	19
4.2.5	Keys Associated with Office Documents.....	20
4.2.6	OpenWithList Key.....	21
4.2.7	ShellBag Keys.....	22
4.3	Metadata Associated with Searching/Browsing	22
4.3.1	Search History	22
4.3.2	TypedURLs Key	23
4.3.3	Favorites Key	25
4.4	Network Related Artifacts found in User Hives	26
4.5	Volume Related Artifacts found in User Hives.....	26
4.6	Computer Metadata Related Artifacts found in User Hives	26

4.7	Persistence Related Artifacts found in User Hives.....	27
5	System Hive Artifacts	27
5.1	Timezone.....	27
5.2	Devices	28
5.3	Shimcache	29
5.4	Computer Related Artifacts found in System Hives.....	30
5.5	Network Related Artifacts found in System Hives	31
5.6	System Restore Related Artifacts found in System Hives	32
5.7	Services Related Artifacts found in System Hives.....	33
5.8	Persistence Related Artifacts found in System Hives	33
5.9	Background/Desktop Activity Moderator	34
5.10	All System artifacts.....	34
6	Software Hive Artifacts	35
6.1	Operating System.....	35
6.2	Class Identifiers (CSLIDs)	35
6.3	In Process Servers (InProcServers).....	36
6.4	Codecs	36
6.5	Desktop related keys (Explorer).....	36
6.6	Installed Software	37
6.7	EmdMgmt	37
6.8	Shell Spawning of an Application.....	37
6.9	Run key Related Artifacts found in Software Hives	37
6.10	Network Related Artifacts found in Software Hives	38
6.11	Volume Related Artifacts found in Software Hives.....	39
6.12	System Restore Related Artifacts found in Software Hives	39
6.13	Web Browsing Related Artifacts found in Software Hives.....	39
6.14	Service Related Artifacts found in Software Hives	39
6.15	Persistence Related Artifacts found in Software Hives.....	40
6.16	All Software artifacts.....	40
7	Security hive Artifacts	41
8	Sam Hive Artifacts	41
9	AmCache Hive Artifacts.....	42

10	Carving Keys from hives [experimental]	43
10.1	Carving keys	43
10.2	Carving Values.....	43
10.3	Hive Statistics	45
11	Merging Transactional Log files into its Associated Hive (Experimental)	46
12	Comparing Hives – Experimental	47
13	Scripting <i>cafae</i> – Experimental	48
13.1	Command line Quick Parse of any Key/Value – Experimental	49
13.2	User Defined Templates (or <i>cmdfiles</i>) - Experimental	52
13.2.1	Template Rules.....	53
13.2.2	Mapping Template parameters to Log2Timeline Output	53
13.2.3	Template Examples.....	54
14	Converting Segmented CSV formats into Database Friendly Formats	55
15	Handling Corrupt Hives	55
16	List of Options	56
17	Authentication and the License File.....	59
17.1	<i>Limited</i> versus <i>Demo</i> versus <i>Full</i> in the tool’s Output Banner	60
18	References	60

TZWorks® Registry Parser (*cafae*) Users Guide

Copyright © TZWorks LLC

Webpage: http://www.tzworks.com/prototype_page.php?proto_id=19

Contact Information: info@tzworks.com

1 Introduction

cafae, which is short for **C**omputer **A**ccount **F**orensic **A**rtifact **E**xtractor, is a Windows registry parser that targets specific registry keys that help identify user activity as it pertains to files and program execution. Chosen are a handful of registry entries that are specific to an account's registry hive(s). This includes both a user's ntuser.dat hive and the usrclass.dat hive for Vista and later. Collectively, these two registry hives contain artifacts useful in piecing together some sort of file/program activity that occurred on a specific account.

Why build another Windows registry parser when there are plenty of good registry parsers freely available on the Internet? The answer is simple. We listened to the feedback that was submitted to our shop by the forensics community; specifically, to take some of the **yaru** functionality^[1] and make an easy to use command line tool. The desire was to be able to use it in a batch processing mode while outputting the data into one of the more common formats so that it could be 'somewhat easily' fused together with varying artifacts from other sources. Prior to v0.17, we focused on ntuser.dat and usrclass.dat hives. Starting with v0.17, we extended report generation to include software, system and security hives.

cafae consists of the same parsing engine that is in **yaru**, but it is packaged into a console application. Consequently, the reports that are generated will look similar to those of **yaru's**, but will have more output options, such as a couple of CSV variants, the ability to change the date format and to set the time precision, among others.

Other useful aspects of **cafae** include the following:

- a. Can parse hives from a live system (same as **yaru**).
- b. Is ubiquitous across *WinXP* through *Win8* (meaning it figures what version of the hive it is working on, and then automatically adjusts which registry keys should be used).
- c. In some cases, it can parse deeper into the metadata and pull out additional artifacts than current registry parsers available.
- d. Non-Windows versions are available for those that choose to process Windows artifacts on a non-Windows operating system.
- e. The architecture is extremely extensible to include additional registry subkeys.

2 Location of Hives

Some of the more common registry hives can be found in the following locations:

Hive	Location
Ntuser.dat	%userprofile%\ntuser.dat
UsrClass.dat	(xp) %userprofile%\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat (vista and later) %userprofile%\AppData\Local\Microsoft\Windows\UsrClass.dat
System	%systemroot%\system32\config\system
Sam	%systemroot%\system32\config\sam
Software	%systemroot%\system32\config\software
Security	%systemroot%\system32\config\Security
Components	(vista and later) %systemroot%\system32\config\Components
BCD	(vista and later) %systemdrive%\boot\bcd
Syscache.hive	(vista and later) System Volume Information\ Syscache.hive
Schema.dat	%systemroot%\System32\SMI\Store\Machine\schema.dat
AmCache.hve	(win 8 and later) %systemroot%\AppCompat\Programs\AmCache.hve
ELAM	(win8 and later) %systemroot%\system32\config\elam
BBi	(win8 and later) %systemroot%\system32\config\bbi
DRIVERS	(win8 and later) %systemroot%\system32\config\drivers

3 How to use *cafae*

cafae is a console application that targets user registry hives (eg. ntuser.dat and usrclass.dat files), as well as software, system and security hives. To use this tool on a live system, one will need to open the command prompt with administrator privileges. One can display the menu options by typing in the executable name without parameters. A screen shot of the menu is shown below. The menu groups the available artifacts by area of analysis, where each artifact can be extracted independently, or combined per command issued. If an artifact comes from a specific type of user hive, the option will identify which hive (either ntuser.dat or usrclass.dat) it expects to receive as input for it to successfully extract data.

```

Usage:
cafae -hive <path> [reg artifacts] [format opts] [misc opts]
cafae -pull_hashes -sam <hive> -system <hive> = *** standalone option

[user hive artifacts - associated with running a program]
-openrun_mru
-userassist
-programs_cache
-muicache = ntuser.dat (xp), usrclass.dat (win7)
-otherapps_run

[ntuser.dat hive artifacts - associated w/ opening files]
-recent_docs
-stream_mru
-opensave_mru
-office_docs
-open_with

[system hive artifacts]
-timezone
-devices = USER
-shimcache = AppCompatCache artifacts
-bam = Backgrd Activity Moderator artifacts

[software hive artifacts]
-clsid = (takes awhile to run)
-inprocservers = Objects invoking DLLs (takes awhile to run)
-codecs
-explorer
-installed_sw
-emdmgmt
-shell_spawn

[Applies to multiple hives]
-runkeys = software and user hives
-computer = system, software and user hives
-network = system, software and user hives
-volumes = sys
-restore = sys
-web = software and user hives
-services = system and software hives
-persistence = system, software and user hives

[collection of registry artifacts]
-all_user = all ntuser and usrclass artifacts
-all_software = all Software artifacts (minus clsid)
-all_system = all System artifacts
-all_security = all
-all_amcache = all
-all_sam = all sam artifacts
-all_syscache = all Syscache artifacts

Basic options (see readme for complete list)
-csv = output in comma separated value format
-csv12t = log2timeline output
-bodyfile = sleuthkit output
-base10 = output number in base10 vice std:hex output
-dateformat mm/dd/yyyy = "yyyy-mm-dd" is the default
-timeformat hh:mm:ss = "hh:mm:ss.xxx" is the default
-pair_datetime = *** combine date/time into 1 field for csv
-no_whitespace = remove whitespace between csv delimiter
-csv_separator "|" = use a pipe char for csv separator
-pipe = pipe hives into cafae for processing
-filter <partial*|*.ext> = *** filters stdin data from -pipe option
-all_controlsets = *** process all ControlSets in System hive

Misc processing options [***]
-stats = ***
-scan_size <min size> = ***
-scan_entropy [80 - 90] = ***
-carve [-vals] = ***
-carve_deleted [-vals] = ***
-merge "log1|log2" -hive <original hive> = ***
-diff "hive1|hive2" = ***

Exposing the internal scripting options [***]
-key <path> [-enumvalues] [-level <#>] [-extract ""] = ***
-cmdfile <filename> [-show null results] = file v

```

The user registry artifacts are grouped into various categories, depending on what is of interest. The report generation for the operating system hives (e.g. *software*, *system*, *sam*, *security*, etc) are all-in-one type options. This means that they pull all the predefined parsers embedded into *cafae* into one report. The output options include: (a) the default output, where each record is on a separate line and each field is separated by the pipe character, (b) the *SleuthKit* body-file format ^[7] and (c) the *log2timeline* CSV (comma separated value) format. ^[8]

To process a specified user hive, one uses the **-hive <location of hive to process>** option with the specific artifact one is interested in. For example, assuming a user hive was extracted to the *c:\dump* directory, one could issue the following command to review all the Microsoft Office documents accessed.

```
cafae -hive c:\dump\ntuser.dat -office_docs > office_docs.txt
```

Since the output that is generated is very wide, it is recommended that one redirect the output of the command into a file as show above. Then, it can be reviewed in any text editor by turning off the word wrap option to view each record on a separate line.

3.1 Volume Shadow Copies

For starters, to access Volume Shadow copies, one needs to be running with administrator privileges. Also, Volume Shadow copies, as is discussed here, only applies to Windows Vista, Win7, Win8 and beyond. It does not apply to Windows XP.

To make it easier with the syntax, we've built in some shortcut syntax to access a specified Volume Shadow copy, via the **%vss%** keyword. This internally gets expanded into `\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy`. Thus, to access index 1 of the volume shadow copy, one would prepend the keyword and index, like so, **%vss%1** to the normal path of the hive. For example, to access a user hive located in the *testuser* account from the *HarddiskVolumeShadowCopy1*, the following syntax can be used:

```
cafae -hive %vss%1\Users\testuser\ntuser.dat -office_docs > office_docs.txt
```

To determine which indexes are available from the various Volume Shadows, one can use the Windows built-in utility **vssadmin**, as follows:

```
vssadmin list shadows
```

To filter some of the extraneous details out, type

```
vssadmin list shadows / find /i "volume"
```

While the amount of data can be voluminous, the keywords one needs to look for are names that look like this:

```
Shadow Copy Volume: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1  
Shadow Copy Volume: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy2  
...
```

From the above, notice the number after the word *HarddiskVolumeShadowCopy*. It is this number that is appended to the **%vss%** keyword.

4 Example User Hive Artifacts that are Extracted and Parsed

As a disclaimer, this version of **cafae** does not contain all the requisite registry keys that may be of interest to a computer forensic analyst, but it does encompass some of the more common ones. Furthermore, every attempt has been made to ensure that this tool parses data correctly; however, there may be cases where the parsing of the data fails. If this happens, please report it to our staff at: info@tzworks.com.

Below are examples of some of the registry artifacts **cafae** can extract. Each subsection includes which registry keys are examined, the command line syntax that was used, and when available, a sample output with annotations.

4.1 Metadata Associated with Running a Program

This is a collection of registry keys that show which program, or application, was used by this user account. If a file was being viewed or edited during this process, the details of the file are listed with this collection of registry keys.

4.1.1 Open -> Run Dialog

- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedMRU`
- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedPidlMRU`
- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedPidlMRULegacy`

These keys track the program that was last used to access the files listed in the Open/Save dialog box MRU subkey. With Vista and later, most of these entries record the timestamp of when the program was executed. The MRU (Most Recently Used) value will show the order of the entries, from the most recently used. **cafae** makes use of this MRU value to sort the output from most to least recently accessed.

Example: `cafae -hive user.win7.hive -openrun_mru > out.txt`

Artifact: Last Visited (Open/Run) PidL MRU
Registry key: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedPidLMRU

reg date	regtime[UTC]	item#	mdate	mtime [UTC]	name	support name
05/10/2012	18:02:23.821	2	05/10/2012	14:46:20	devenv.exe	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	19	05/08/2012	19:24:30	{9c5a20b3-2227-4506-b205-0f00cda5554d}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	20	05/08/2012	19:24:30	{9c5a20b3-2227-4506-b205-0f00cda5554d}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	5	05/01/2012	16:14:28	{2d28d28}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	1	05/08/2012	16:14:28	{2d28d28}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	9	05/03/2012	20:52:34	{E80B43BA-65A2-48A3-B401-036D7BF26319}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	12	05/03/2012	20:12:28	{A23C542E-7E25-4896-A5E6-108EF4286F53}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	23	05/03/2012	20:12:28	{B5E83989-4076-4ED0-A33E-988E9870B07F}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	10	04/23/2012	14:57:34	hworks64.exe	{CLSID_MyComputer}\E:\tutorials\
05/10/2012	18:02:23.821	7	04/22/2012	16:42:50	NOTEPAD.EXE	{CLSID_MyComputer}\G:\yaru.analy
05/10/2012	18:02:23.821	8	04/16/2012	17:59:44	ultraISO.exe	{CLSID_MyComputer}\w:\wd\public\
05/10/2012	18:02:23.821	4	04/05/2012	20:00:32	iexplore.exe	{CLSID_MyComputer}\w:\wd\tzlabs\
05/10/2012	18:02:23.821	6	03/31/2012	14:02:04	{5B18016A-C18C-4D3E-BF41-67F13402487F}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	24	03/27/2012	14:22:48	{78BD588C-652B-40C7-AB74-E231A23348E8}	{CLSID_MyComputer}\w:\wd\tzlabs\
05/10/2012	18:02:23.821	3	03/18/2012	16:57:46	{F34A085E-7805-49C3-8EA3-3483D6D17F6F}	{CLSID_MyComputer}\C:\Users\tzla
05/10/2012	18:02:23.821	0	03/01/2012	19:20:28	appverif.exe	{CLSID_MyComputer}\E:\tutorials\
05/10/2012	18:02:23.821	17	01/29/2012	18:18:28	{FFA2B03A-9881-47D6-9C45-54541F3865C1}	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	21	02/02/2012	16:14:40	mspaint.exe	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	22	02/02/2012	15:23:46	SnagitEditor.exe	{CLSID_MyComputer}\E:\class_v2\w
05/10/2012	18:02:23.821	11	09/11/2011	12:34:26	rundll32.exe	{CLSID_MyComputer}\C:\Program Fi
05/10/2012	18:02:23.821	13	05/27/2011	21:43:18	{47602FAA-4542-4F08-A904-029314D8CD0B}	{CLSID_MyComputer}\G:\business.c
05/10/2012	18:02:23.821	18	12/05/2011	21:32:18	{F96CCFA6-EE86-40BD-A7D1-299F49D11C62}	{CLSID_MyComputer}\w:\wd\tzlabs\
05/10/2012	18:02:23.821	16	12/03/2011	18:38:34	{663EAAE3-ACCB-478F-AC19-ECA2536E0C64}	{CLSID_MyComputer}\w:\wd\public\
05/10/2012	18:02:23.821	15	11/03/2011	18:41:36	Procmon64.exe	{CLSID_MyComputer}\E:\win7
05/10/2012	18:02:23.821	14	10/28/2011	01:21:00	windbg.exe	{CLSID_MyComputer}\C:\dump\hide

For reference purposes, below is a screenshot of the raw data for item #4 (iexplore.exe) as viewed with **yaru**. One can see there is some readable data in the hexadecimal output, but forming a complete path to the application as well as extracting temporal metadata requires some additional parsing.

yaru - (version 1.18) (C:\dump\cafae.test\user.win7.hive)

File View Options Reports

Registry Keys

ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedPidLMRU\4

REG_BINARY

Raw data from item #4 (as viewed by yaru)

0000 0000: 69 00 65 00 78 00 70 00 6c 00 6f 00 72 00 65 00 i.e.x.p.l.o.r.e.
0000 0010: 2e 00 65 00 78 00 65 00 00 00 14 00 1f 50 e0 4f ..e.x.e....P.O
0000 0020: d0 20 ea 3a 69 10 a2 d8 08 00 2b 30 30 9d 19 00 .i...+00...
0000 0030: 2f 57 3a 5c 00 00 00 00 00 00 00 00 00 00 00 00 /W:.....
0000 0040: 00 00 00 00 00 00 00 00 00 00 44 00 31 00 00 00 24D.1...\$
0000 0050: 3f 01 04 20 00 72 64 00 00 22 00 08 00 04 00 ef ?..0.vd..2...
0000 0060: be 00 00 00 00 00 00 00 00 00 ad 34 00 ..>lm=?..e*..4.
0000 0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..w.d...P.1..
0000 0080: 00 00 00 00 00 00 00 00 00 00 50 00 31 00 00 00 f@"..0.tzlabs..
0000 0090: 00 00 00 00 00 00 00 00 00 00 a 6c 61 62 73 00 ..t.....nf@..
0000 00a0: 00 3a 00 08 00 04 00 e1 be a1 3e 1a 6d 66 40 22 ..*...t.....t.z.1
0000 00b0: 8c 2a 00 00 00 74 2e 00 00 00 00 01 00 00 00 00 ..a.b.s...X.1..
0000 00c0: 00 00 00 00 00 00 00 00 00 00 16 00 58 00 31 00 00 ..le...COMPAN~
0000 00d0: 00 00 00 7c 40 ea 98 10 00 43 4f 4d 50 41 4e 7e ..e 72 4f 7e ..
0000 00e0: 31 00 00 40 00 98 00 04 ..e 72 4f 7e ..

4.1.2 UserAssist Key

- ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist

This key contains values that identify programs executed by a user account. Entries are a mix of executable files and an associated link entry. Many of the entries contain the last execution time along with the number of times the application was run. While the last execution time seems reliable, the run count data is still under evaluation and is based on empirical data. Therefore the results of this output should be considered *experimental*.

Example: *cafae -hive user.win7.hive -userassist > out.txt*

Artifact: UserAssist
Registry key: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist*\Count

regdate	regtime [UTC]	mod date	mtime [UTC]	sess	run	subkey	item name
05/10/2012	18:57:09.323	05/10/2012	18:42:04.796	17	1	{CEBFF...}	{F38BF404-1D43-42F2-9305-67DE0B28FC23}\regedit.exe
05/10/2012	18:57:09.323	05/10/2012	18:28:05.728	17	4	{CEBFF...}	C:\tools\home\varu64.exe
05/10/2012	18:57:09.323	05/10/2012	18:15:48.670	17	97	{CEBFF...}	...
05/10/2012	18:57:09.323	05/10/2012	18:15:48.670	17	14	{F4E57...}	...
05/10/2012	18:57:09.323	05/10/2012	18:03:38.764	17	58	{CEBFF...}	...
05/10/2012	18:15:48.670	05/10/2012	18:03:38.764	17	58	{F4E57...}	...
05/10/2012	18:57:09.323	05/10/2012	18:02:13.544	17	57	{CEBFF...}	visu...
05/10/2012	18:15:48.670	05/10/2012	18:02:13.544	17	57	{F4E57...}	...
05/10/2012	18:57:09.323	05/10/2012	17:41:59.110	17	21	{CEBFF...}	7C5...
05/10/2012	18:15:48.670	05/10/2012	17:41:59.110	17	21	{F4E57...}	...
05/10/2012	18:57:09.323	05/10/2012	17:28:49.482	17	9	{CEBFF...}	7C5...
05/10/2012	18:57:09.323	05/10/2012	16:47:17.371	17	18	{CEBFF...}	1AC...
05/10/2012	18:57:09.323	05/10/2012	16:47:17.371	17	18	{F4E57...}	...
05/10/2012	18:57:09.323	05/10/2012	15:58:11.907	17	7	{CEBFF...}	9E3995AB-1F9C-4F13-B827-48B24B6C7174}\TaskBar\Com...
05/10/2012	18:57:09.323	05/10/2012	15:55:27.618	17	31	{CEBFF...}	6D809377-6AF0-4448-8957-A3773F02200E}\Windows NT\...
05/10/2012	18:15:48.670	05/10/2012	15:28:10.560	17	1	{F4E57...}	7C5A40EF-A0FB-4BFC-874A-C0F2E089FA8E}\Internet Ex...
05/10/2012	18:57:09.323	05/10/2012	15:28:10.559	17	1	{CEBFF...}	9E3995AB-1F9C-4F13-B827-48B24B6C7174}\TaskBar\Mid...
05/10/2012	18:15:48.670	05/10/2012	12:47:45.188	17	1	{CEBFF...}	7C5A40EF-A0FB-4BFC-874A-C0F2E089FA8E}\Microsoft V...
05/10/2012	18:57:09.323	05/10/2012	12:47:45.188	17	1	{CEBFF...}	VMware.Workstation.vmui
05/10/2012	18:15:48.670	05/10/2012	12:47:45.188	17	1	{CEBFF...}	9E3995AB-1F9C-4F13-B827-48B24B6C7174}\TaskBar\VMw...
05/10/2012	18:57:09.323	05/10/2012	12:16:35.938	17	1	{CEBFF...}	7C5A40EF-A0FB-4BFC-874A-C0F2E089FA8E}\Common File...

Use of a wildcard to traverse the 'new' unknown subdirectory keys which vary with operating system

Last time run

Times run

Below is a view of the raw data of the first entry above. One can see the names are scrambled in the native form and require a form of ROT-13 (rotate by 13 places) unscrambling to de-obfuscate the data. The listing order is changed in the final output above and is based on most recent modify date.

Registry Keys	Key/Value data
P:\gbbbyf\ubzr\ngk_ivrb4.rkr	ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\
R:\pydff_i2\jva32_cebwrpgf\rybtivj\jva32\Qroht\ngk_ivrb4.rkr	{S380S404-1Q43-42S2-9305-67QR0028SP23}\ertrqvg.rkr REG_BINARY
R:\pydff_i2\jva32_cebwrpgf\hgvyf\jva32\Qroht\hgvyf.rkr	0000 0000: 11 00 00 00 01 00 00 00 01 00 00 00 02 07 00 00
{S380S404-1Q43-42S2-9305-67QR0028SP23}\ertrqvg.rkr	0000 0010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
{7P5N40RS-N0SO-4OSP-874N-P0S2R0O9SN8R}\GruFzvgu\Pnzg	0000
{7P5N40RS-N0SO-4OSP-874N-P0S2R0O9SN8R}\GruFzvgu\Pnzg	0000
R:\pydff_i2\jva32_cebwrpgf\ubzr\ngk_ivrb4.rkr	0000

Scrambled name equates to {F38BF404-1D43-42F2-9305-67DE0B28FC23}\regedit.exe

4.1.3 ProgramsCache Key

- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage\ProgramsCache*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage2\ProgramsCache*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage2\ProgramsCache TBP*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage2\ProgramsCacheSMP*

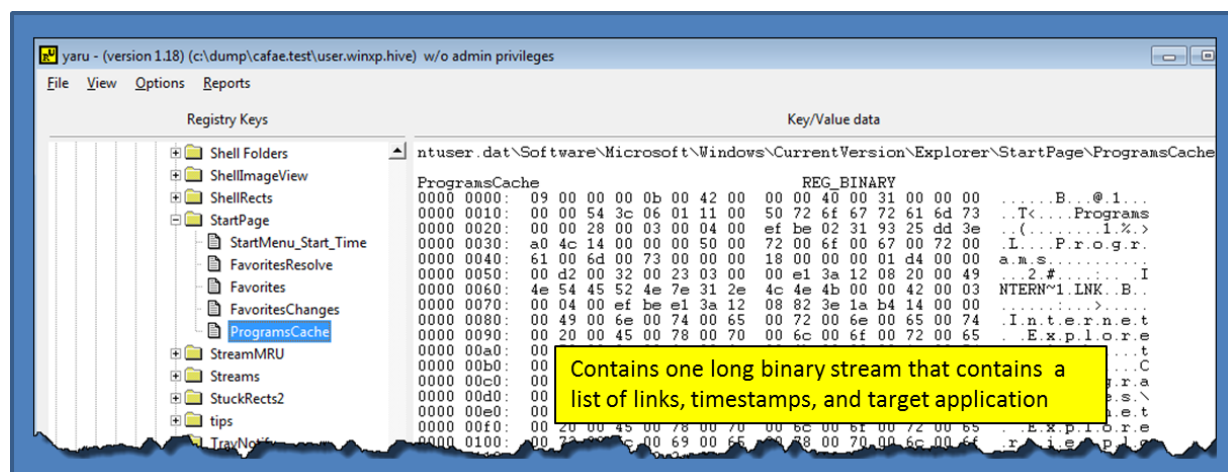
The *ProgramsCache* key records which application was launched as well as when it was launched. The size of the file refers to the link file, and not the target file the link points to. Not shown in the diagram below, but included in the truncated section on the right, are additional parameters that may have been present in the data.

Example: `cafae -hive user.winxp.hive -programs_cache > out.txt`

Artifact: ProgramsCache
Registry key: NTUSER.DAT\Software\Microsoft\windows\CurrentVersion\Explorer\StartPage\ProgramsCache

reg date	regtime[UTC]	mdate	t		am/link
02/20/2011	17:04:13.640	02/20/2011	17:33:26	02/20/2011	Programs
02/20/2011	17:04:13.640	02/20/2011	17:33:26	02/20/2011	Programs\Internet Explorer.lnk
02/20/2011	17:04:13.640	02/20/2011	17:33:20	02/20/2011	Programs\Outlook Express.lnk
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Remote Assistance.lnk
02/20/2011	17:04:13.640	02/20/2011	17:33:22	02/20/2011	Programs\Windows Media Player.lnk
02/20/2011	17:04:13.640	02/20/2011	17:33:22	02/20/2011	Programs\Accessories
02/20/2011	17:04:13.640	02/20/2011	17:33:22	02/20/2011	Programs\Accessories
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Windows Explorer.lnk
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Accessibility
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Accessibility\Magnifier.lnk
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Accessibility\Narrator.lnk
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Accessibility\On-Screen Reader.lnk
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Accessibility\Utility Manager.lnk
02/20/2011	17:04:13.640	02/20/2011	17:18:56	02/20/2011	Programs\Accessories\Accessibility\Windows Narrator.lnk

The raw *ProgramsCache* registry entry contains one very large binary chunk of data that contains the links, target applications associated with the links, the timestamps for each link and any parameters that point to an icon.



4.1.4 MUICache Key

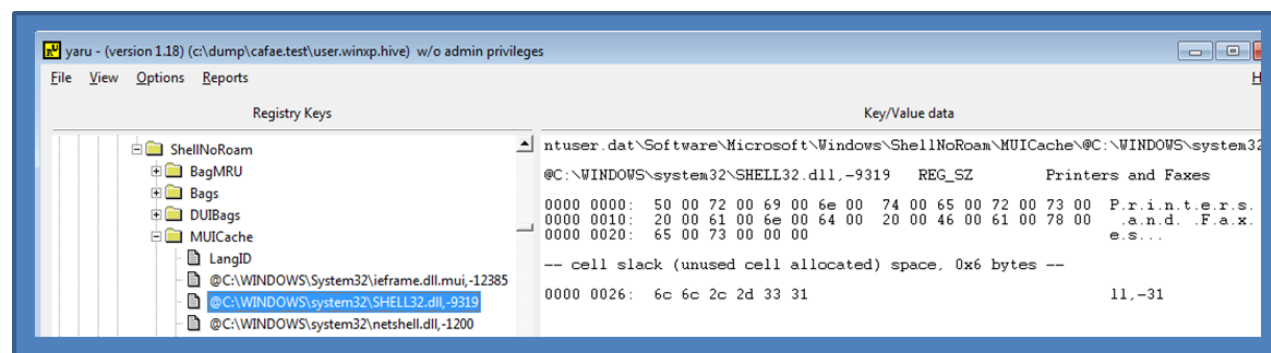
- *ntuser.dat\Software\Microsoft\Windows\ShellNoRoam\MUICache*
- *usrclass.dat\Local Settings\MuiCache*
- *usrclass.dat\Local Settings\Software\Microsoft\Windows\Shell\MuiCache*

The operating system records what applications are launched by a particular user account. The *MUIcache* subkey records the *name* of the application and the *File Description* information. This description is taken from the version information stored in the portable executable of the binary that was launched. Unfortunately, there is no execution timestamp information associated with each entry.

Example: *cafae -hive user.winxp.hive -muicache > out.txt*

Artifact: MUICache			
Registry key: NTUSER.DAT\Software\Microsoft\windows\ShellNoRoam\MUICache			
reg date	regtime[UTC]	value name	value data
05/12/2012	23:05:56.061	LangID	09 04
05/12/2012	23:05:56.061	C:\WINDOWS\Explorer.EXE	Windows Explorer
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\wiasext.dll,-331	Scanners and Cameras
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\usmt\migwiz.exe,-202	Files and Settings Transfer wizard
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\tourstart.exe,-1	Tour windows XP
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\spider.exe,-56	Spider Solitaire
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\restore\rstrui.exe,-2048	System Restore
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\rcbdyct1.dll,-152	Remote Assistance
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\oobe\msOOBE.exe,-2000	Activate windows
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\netshell.dll,-1200	Network Connections
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\netshell.dll,-1010	New Connection wizard
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\mshearts.exe,-413	Hearts
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\hnetwiz.dll,-3085	Network Setup wizard
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\compatUI.dll,-115	Program Compatibility wizard
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\SHELL32.dll,-9319	Printers and Faxes
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\SHELL32.dll,-9227	My Documents
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\SHELL32.dll,-9217	My Network Places
05/12/2012	23:05:56.061	@C:\WINDOWS\system32\SHELL32.dll,-9216	My Computer

The raw data for the *MUICache* is rather is straight forward to parse. Each entry is a separate value name/data pair. The value data is in UTF-16 format.



4.1.5 Run Key and Miscellaneous Applications

- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Run*
- *ntuser.dat\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Run*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\RunOnce*
- *ntuser.dat\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\RunOnce*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\RunServices*
- *ntuser.dat\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\RunServices*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run*
- *ntuser.dat\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\Terminal Server\Install\Software\Microsoft\Windows\CurrentVersion\Run*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\Terminal Server\Install\Software\Microsoft\Windows\CurrentVersion\Runonce*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\Terminal Server\Install\Software\Microsoft\Windows\CurrentVersion\RunonceEx*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Policies\System\Shell*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\Load*
- *ntuser.dat\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Load*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU*
- *ntuser.dat\Software\Microsoft\IntelliPoint\AppSpecific*
- *ntuser.dat\Software\Sysinternals*

The *Run* and *RunOnce* keys cause programs to run when a user logs on. It is a common and well known way for applications to remain persistent across reboots.

For this example, however, we highlighted the *SysInternals* subkey along with its *EulaAccepted* value. If the value is set to a 1, then the user accepted the EULA and the tool ran. This is a good way to tell which *SysInternals* tools have been run on the system.

Example: *cafae -hive user.win7.hive -runkeys > out.txt*

Example: *cafae -hive user.win7.hive -otherapps_run > out.txt*

Artifact: SysInternals Apps Run				
Registry key: NTUSER.DAT\Software\Sysinternals\?*\.EulaAccepted				
reg date	regtime[UTC]	subkey	value name	value data
01/24/2012	22:48:11.984	Process Explorer	EulaAccepted	1
12/20/2011	22:41:17.793	AutoRuns	EulaAccepted	1
11/09/2011	16:47:47.444	Strings	EulaAccepted	1
11/03/2011	20:38:06.438	Process Monitor	EulaAccepted	1
10/31/2011	19:53:57.998	TCPView	EulaAccepted	1
09/15/2011	01:08:06.636	psfile	EulaAccepted	1

SystInternals tools run by this user account

Artifact: Run Values				
Registry key: NTUSER.DAT\Software\Microsoft\windows\CurrentVersion\Run				
reg date	regtime[UTC]	value name	value data	
09/11/2011	12:01:43.284	Sidebar	C:\Program Files\windows sidebar\sidebar.exe /autoRun	Run key
Artifact: Last Command Executed				
Registry key: NTUSER.DAT\Software\Microsoft\windows\CurrentVersion\Explorer\RunMRU				
reg date	regtime[UTC]	item#	filename	
10/02/2012	15:36:05.533	-> a	explorer\1	Last command executed

The raw data for this set of subkeys is straight forward to parse. Each entry is a separate value name/data pair. All the value data is either a flag (1 or 0), or straight null terminated strings.

4.1.6 FeatureUsage

- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\FeatureUsage\AppBadgeUpdated*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\FeatureUsage\AppLaunch*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\FeatureUsage\AppSwitched*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\FeatureUsage\ShowJumpView*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\FeatureUsage\TrayButtonClicked*

These set of subkeys track events associated with the Task Bar when a user has an application pinned to it. Many of the subkeys are meant to be self-explanatory in their reporting metric. For example:

- *AppBadgeUpdated* provides the number of times an application had badge updates on the taskbar.
- *AppLaunch* shows the number of times an application is launched from the taskbar.
- *AppSwitch* shows the number of times an application switched focus.
- *ShowJumpView* shows the number of times an application was right-clicked on the taskbar
- *TrayButtonClicked* shows the number of times the built-in taskbar buttons were clicked.

4.2 Metadata Associated with Viewing/Opening/Editing Files

This is a collection of registry keys that show some metadata of the file viewed or edited.

4.2.1 Recent Documents and Associated Keys

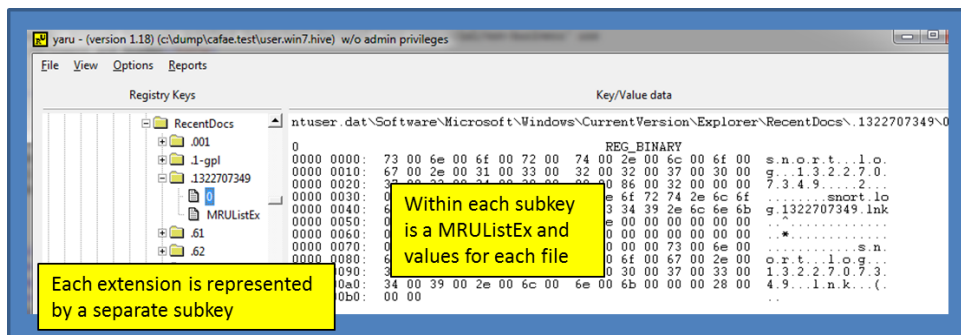
- *ntuser.dat\Software\Adobe\?*?*AVGeneral\cRecentFiles*
- *ntuser.dat\Software\America Online\AOL Instant Messenger (TM)\CurrentVersion\Users*
- *ntuser.dat\Software\BreakPoint\?**\Recent File List*
- *ntuser.dat\Software\Microsoft\Dependency Walker\Recent File List*
- *ntuser.dat\Software\Microsoft\MediaPlayer\Player\RecentFileList*
- *ntuser.dat\Software\Microsoft\Microsoft Management Console\Recent File List*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Applets\?**\Recent*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Applets\?**\Recent File List*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Applets\Regedit*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Applets\RegEdit\Favorites*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Search\RecentApps*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Search\JumplistData*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\TypedPaths*
- *ntuser.dat\Software\Nico Mak Computing\WinZip\extract*
- *ntuser.dat\Software\Nico Mak Computing\WinZip\mru\archives*
- *ntuser.dat\Software\Symantec\Symantec Ghost\?**\Recent File List*
- *ntuser.dat\Software\WinRAR\ArchHistory*

For the standard 'Explorer\RecentDocs' item, this key contains the recent documents as identified in the Windows "My Recent Documents" menu. Within the key is a *MRUListEx* value that identifies the most recently viewed items. If one parses the *MRUListEx*, one can display the items in the order that they were accessed relative to each other. *cafae* will output this list of items in the proper order starting with the most recently viewed first. While there is no temporal information associated with each entry, one can use registry last modification time associated with the subkey to determine when the most recent item was opened.

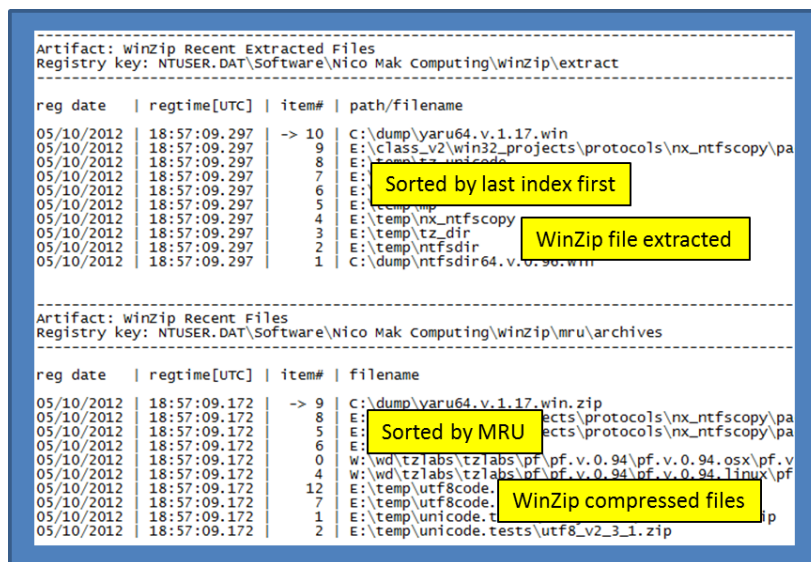
Example: *cafae -hive user.win7.hive -recent_docs > out.txt*

reg date	regtime[UTC]	item#	subkey	name
05/10/2012	18:57:09.216	-> 2	.zip	
05/10/2012	18:57:09.216	7	.zip	
05/10/2012	18:57:09.216	5	.zip	
05/10/2012	18:57:09.216	3	.zip	
05/10/2012	18:57:09.216	4	.zip	nxx64.v.0.07.zip
05/10/2012	18:57:09.216	0	.zip	nxx32.v.0.07.zip
05/10/2012	18:57:09.216	1	.zip	pf64.v.0.95.win.zip
05/10/2012	18:57:09.216	6	.zip	pf32.v.0.95.win.zip
05/10/2012	18:57:09.216	8	.zip	
05/10/2012	18:57:09.216	9	.zip	
05/10/2012	18:02:23.855	-> 9	.vcxproj	
05/10/2012	18:02:23.855	3	.vcxproj	
05/10/2012	18:02:23.855	0	.vcxproj	
05/10/2012	18:02:23.855	6	.vcxproj	
05/10/2012	18:02:23.855	2	.vcxproj	
05/10/2012	18:02:23.855	4	.vcxproj	
05/10/2012	18:02:23.855	7	.vcxproj	
05/10/2012	18:02:23.855	9	.vcxproj	

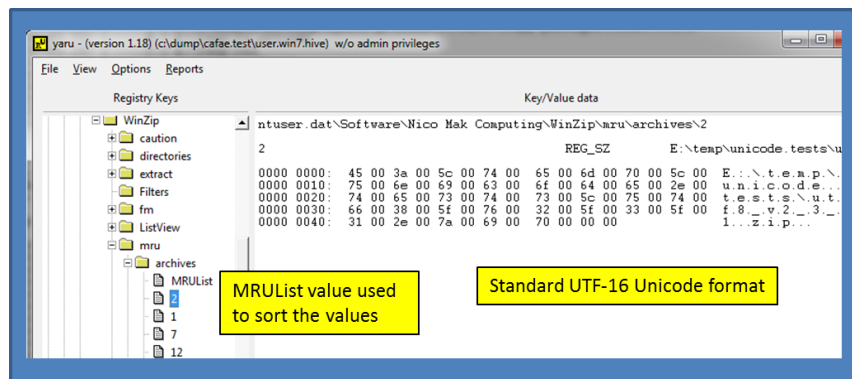
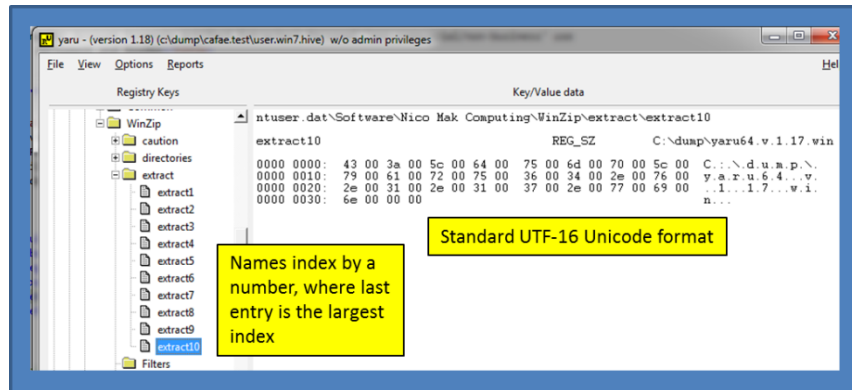
cafae sorts these entries by folder subkey and then by most recently used. The output includes an arrow to designate which item was the last one modified. When looking at the raw entries, one can see each extension represented by a separate subkey and each file is a separate child value within the subkey.



For other keys that this option parses, a variant of the MRU list is used. Some have a MRU value that can be used to sort the proper order, but others just use an index number as part of the name. See the figure below for how WinZip artifacts are displayed.



These entries are straight forward to parse, since the value data is in UTF-16 format, and the value names are either indexed by a MRUList value or by index number in the name to show the most recently used.



4.2.2 JumplistData Key

- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Search\JumplistData`

This is a new key that is available with the Win10 April 2018 update. It can be invoked explicitly with `-jumplistdata` or included with the `-recent_docs` option.

Example: `cafae -hive user.win10.hive -jumplistdata > out.txt`

Example: `cafae -hive user.win10.hive -recent_docs > out.txt`

4.2.3 StreamMRU Key

- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StreamMRU`

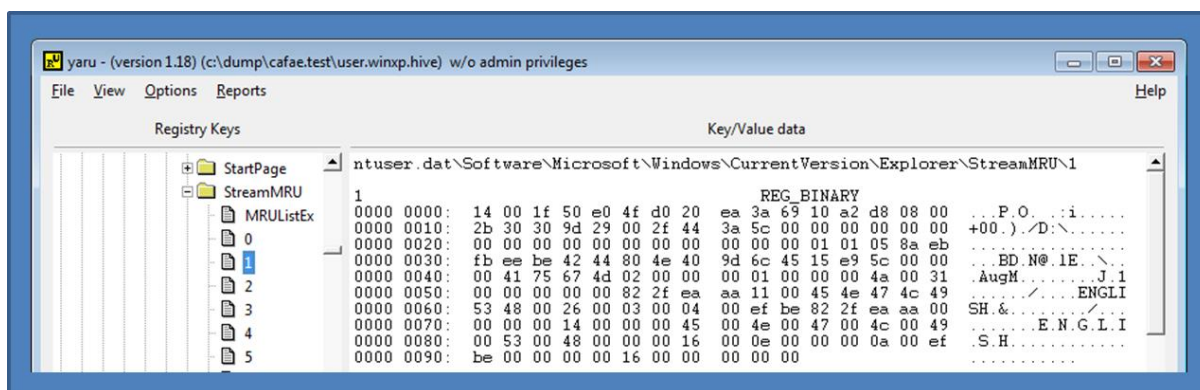
Per the MSDN article 235994, the Streams registry entries store the size and location information for closed windows. The article states that Windows saves this information for up to 28 different windows. The association for the Streams subkey with a particular window is stored in the *StreamMRU* subkey. As one can see there are embedded timestamps for many of the entries.

Example: `cafae -hive user.winxp.hive -stream_mru > out.txt`

Artifact: Stream MRU
Registry key: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\StreamMRU

reg date	regtime[UTC]	item#	mdate	mtime [UTC]	adate	name
05/15/2012	20:33:11.104	-> 138			05/15/2012	{CLSID_MyComputer}\G:
05/15/2012	20:33:11.104	193	03/21/2012	14:06:02	03/21/2012	{CLSID_MyComputer}\G:\copyright.material
05/15/2012	20:33:11.104	189	10/23/2011	14:01:04	10/24/2011	{CLSID_MyComputer}\G:\crash.analysis
05/15/2012	20:33:11.104	0				{CLSID_MyComputer}\D:
05/15/2012	20:33:11.104	188	09/19/2011	02:10:44	09/20/2011	{CLSID_MyComputer}\G:\hw1
05/15/2012	20:33:11.104	186	09/25/2011	19:11:28	09/28/2011	{CLSID_MyComputer}\G:\new.stuff\forensics
05/15/2012	20:33:11.104	185	09/25/2011	15:43:26	09/28/2011	{CLSID_MyComputer}\G:\new.stuff
05/15/2012	20:33:11.104	187	09/25/2011	15:54:46	09/28/2011	{CLSID_MyComputer}\G:\new.stuff\forensics\windows
05/15/2012	20:33:11.104	62				{CLSID_MyComputer}\E:
05/15/2012	20:33:11.104	179	08/15/2005	16:49:28		{CLSID_MyComputer}\E:\SQL Server 2000
05/15/2012	20:33:11.104	178				{CLSID_MyComputer}\I:
05/15/2012	20:33:11.104	173	01/03/2010	23:00:14	06/02/2010	{CLSID_MyComputer}\G:\tools
05/15/2012	20:33:11.104	174				{CLSID_MyComputer}\H:
05/15/2012	20:33:11.104	177	01/12/2011	21:51:48	01/12/2011	{CLSID_MyComputer}\H:\case1
05/15/2012	20:33:11.104	176	01/02/2011	15:52:42	01/04/2011	{CLSID_MyComputer}\H:\reversing
05/15/2012	20:33:11.104	170				{CLSID_MyComputer}\G:
05/15/2012	20:33:11.104	169	04/29/2008	23:26:56	07/24/2008	{CLSID_MyComputer}\G:\
05/15/2012	20:33:11.104	168	06/25/2008	00:54:32	06/24/2008	{CLSID_MyComputer}\G:\
05/15/2012	20:33:11.104	167	06/28/2009	18:09:12	07/14/2009	{CLSID_MyComputer}\G:\
05/15/2012	20:33:11.104	166	06/28/2009	18:09:12	06/28/2009	{CLSID_MyComputer}\G:\
05/15/2012	20:33:11.104	165	03/17/2009	23:05:28	04/07/2009	{CLSID_MyComputer}\G:\

The *StreamMRU* data has a value per entry. Each entry is not straight forward to read with just a hex editor and requires custom parsing.



4.2.4 Open -> Save Dialog

- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSaveMRU`
- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSavePidlMRU`

The operating system tracks files that have been opened or saved, from an "Open/Save As" shell dialog box through these registry keys. They contain multiple subkeys that represent different extensions. Each subkey that represents an extension contains values sorted by a MRU list. With Vista and later, most of these entries record the timestamp when the action occurred. *cafae* sorts these entries by folder subkey, and then by most recently used. The output includes an arrow to designate which item was the last one modified.

Example: *cafae* `-hive user.win7.hive -opensave_mru > out.txt`

reg date	regtime[UTC]	item#	subkey	mdate	mtime[UTC]	adate	atime[UTC]	cdate	ctime[UTC]	file size	name
05/10/2012	18:02:23.824	-> 11	*	05/09/2012	13:20:08	05/09/2012	13:20:08	03/10/2012	19:13:22	0x000089c1	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	15	*	05/09/2012	13:16:54	05/09/2012	13:16:54	03/10/2012	19:06:22	0x0000557e	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	16	*	05/09/2012	13:12:10	05/09/2012	13:12:10	01/07/2012	13:54:42	0x00004bfb	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	3	*	05/09/2012	12:57:26	05/09/2012	12:57:26	05/09/2012	12:56:12	0x00008b64	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	2	*	05/09/2012	13:17:42	05/09/2012	13:17:42	02/10/2012	21:52:24	0x000019e3c	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	7	*	05/09/2012	13:18:52	05/09/2012	13:18:52	03/10/2012	18:58:08	0x00007470	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	10	*	05/09/2012	13:08:54	05/09/2012	13:07:56	12/29/2011	00:44:02	0x00004ebf	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	17	*	05/09/2012	13:05:28	05/09/2012	13:04:28	01/15/2012	20:41:26	0x00009c2e	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	14	*	05/09/2012	13:09:58	05/09/2012	13:09:14	05/04/2012	12:59:32	0x00004c3f	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	9	*	05/09/2012	13:09:58	05/09/2012	13:09:14	05/04/2012	12:59:32	0x00005a62	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	13	*	05/09/2012	13:09:58	05/09/2012	13:09:14	05/04/2012	12:59:32	0x0000b3f7	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	6	*	05/09/2012	13:09:58	05/09/2012	13:09:14	05/04/2012	12:59:32	0x0000630c	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.824	8	*	05/09/2012	13:06:04	05/09/2012	13:06:04	05/09/2012	13:06:04	0x00003894d	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.823	-> 11	vcxproj	05/09/2012	12:59:08	05/09/2012	12:59:08	05/09/2012	12:59:08	0x000098ab	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.823	19	vcxproj	05/09/2012	12:59:08	05/09/2012	12:59:08	05/09/2012	12:59:08	0x00009ae0	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.823	18	vcxproj	05/09/2012	16:56:28	05/10/2012	19:03:56	03/10/2012	19:03:56	0x00008323	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.823	6	vcxproj	05/07/2012	13:29:08	05/10/2012	15:01:32	03/10/2012	15:01:32	0x000029c1	(CLSID_MyComputer)\E:\win7\...
05/10/2012	18:02:23.823	3	vcxproj	05/07/2012	13:29:08	05/10/2012	15:01:32	03/10/2012	15:01:32	0x000029c1	(CLSID_MyComputer)\E:\win7\...

The raw data of the subkey structure is shown below. One of the entries under the 'dll' subkey is highlighted to show an example of the raw value data and the name it is parsed to.

Key/Value data
REG_BINARY
0000 0000: 14 00 1f 50 e0 4f d0 20 ea 3a 69 10 a2 d8 08 00 ...P.O...i...
0000 0010: 2b 30 30 9d 19 00 2f 45 3a 5c 00 00 00 00 00 ...+00.../E:\...
0000 0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...c?r...win7...
0000 0030: 00 00 00 00 00 63 3f a7 72 10 00 77 69 6e 37 00 ...6...c?rc?...
0000 0040: 00 36 00 08 00 04 00 ef be 63 3f a3 72 63 3f a7 ...r...4...
0000 0050: 72 2a 00 00 00 db 2f 00 00 00 00 34 00 00 00 ...%...
0000 0060: 00 00 00 00 00 00 00 00 00 00 00 77 00 69 00 6e ...x.6.4...F.1...
0000 0070: 00 37 00 00 00 14 00 46 00 31 00 00 00 00 00 63 ...c?r...bin.4...
0000 0080: 3f aa 72 10 00 78 36 34 00 34 00 08 00 04 00 ef ...c?rc?r...*
0000 0090: be 63 3f a7 72 63 3f aa 72 2a 00 00 00 dc 2f 00 ...2...c?...
0000 00a0: 00 00 00 25 00 00 00 00 00 00 00 00 00 00 00 ...7FE5E5~1.DLL...
0000 00b0: 00 00 00 78 00 36 34 00 00 00 12 00 46 00 31 ...c?r...c?...
0000 00c0: 00 00 00 00 00 63 3f b7 80 10 00 62 69 6e 00 34 ...7.f.0.0...
0000 00d0: 00 08 00 04 00 ef be 63 3f a9 72 63 3f b7 80 2a ...0.0.0.1...c.c...
0000 00e0: 00 00 00 dd 2f 00 00 00 00 00 00 69 00 04 00 00 ...9.a.4.1...b.3.f.7...
0000 00f0: 00 00 00 00 00 00 00 00 00 62 00 69 00 00 00 ...1.0.3.0...b.9.b...
0000 0100: 00 12 00 d6 00 32 00 00 be 11 00 63 3f 13 80 20 ...4.2.a.3.0.2.3.2...
0000 0110: 00 39 00 61 00 34 00 31 00 62 00 39 00 66 00 37 ...5.5.3.7.d.7.b.9...
0000 0120: 00 31 00 30 00 33 00 30 00 5f 00 62 00 39 00 62 ...d.c.5.2.d.4.7.f...
0000 0130: 00 34 00 32 00 61 00 33 00 30 00 62 00 39 00 66 ...3.3.a.7.3...k.e...
0000 0140: 00 35 00 35 00 33 00 37 00 64 00 37 00 62 00 39 ...[CLSID_MyComputer]\E:\win7\...
0000 0150: 00 64 00 63 00 35 00 32 00 64 00 34 00 37 00 66 ...
0000 0160: 00 33 00 33 00 61 00 37 00 33 00 5f 00 6b 00 65 ...

4.2.5 Keys Associated with Office Documents

- `ntuser.dat\Software\Microsoft\Office\?*?*File MRU`
- `ntuser.dat\Software\Microsoft\Office\?*?*Place MRU`
- `ntuser.dat\Software\Microsoft\Office\?*?*Recent File List`
- `ntuser.dat\Software\Microsoft\Office\?*?*Security\Trusted Documents\TrustRecords`
- `ntuser.dat\Software\Microsoft\Office\?*Recent Files`
- `ntuser.dat\Software\Microsoft\Office\Common`

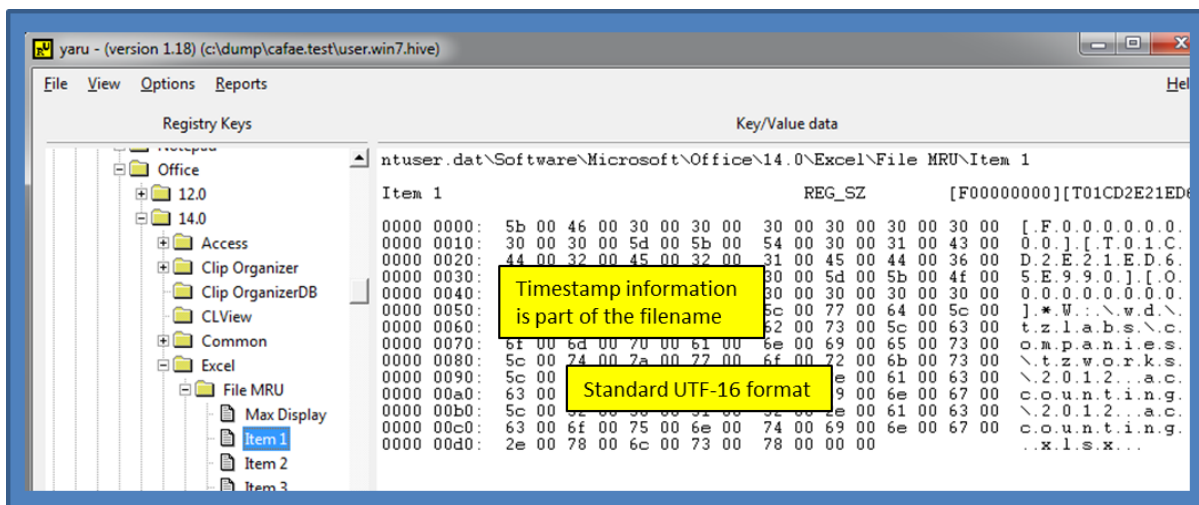
Per the MSDN article 826208, many Microsoft Office programs maintain a list of the most recently used (MRU) files. Additionally, the various Office programs display this MRU list on the File menu and in several other locations. These locations include the *Open* dialog box, the *Save As* dialog box, and the *Insert Hyperlink* dialog box. The purpose of this feature is to provide quick access to files that a user is working on. When extracting this data from the relevant subkeys, one will be able to not only see the file that was used, but the actual timestamp that file was accessed.

Example: `cafae -hive user.win7.hive -office_docs > out.txt`

Artifact: Office File MRU
Registry key: NTUSER.DAT\Software\Microsoft\Office\?*?*File MRU

reg date	regtime[UTC]	open date	time [UTC]	tag	item#	subkey	path/filename
05/10/2012	17:41:07.054	05/10/2012			1	PowerPoint	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/09/2012			1	Excel	W:\wd\tz1abs\companies...
05/09/2012	20:32:47.680	05/08/2012			2	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/08/2012			3	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/08/2012			4	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/08/2012			5	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/08/2012	16:24:49.644	F00000000	6	Excel	C:\windows\system32\out...
05/09/2012	20:32:47.680	05/08/2012	15:42:03.247	F00000000	7	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/08/2012	15:38:29.457	F00000000	8	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/07/2012	13:25:36.938	F00000000	9	Excel	E:\class_v2\win32_proj...
05/10/2012	17:41:07.054	05/05/2012	16:14:51.363	F00000000	2	PowerPoint	C:\dump\cypriot.pptx
05/09/2012	20:32:47.680	05/05/2012	14:01:04.813	F00000000	10	Excel	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/05/2012	8:29:43.540	F00000000	11	Excel	E:\class_v2\win32_proj...
05/03/2012	20:32:47.680	05/03/2012	0:12:58.718	F00000002	1	Word	E:\class_v2\win32_proj...
05/09/2012	20:32:47.680	05/03/2012	0:08:40.495	F00000000	2	Word	W:\wd\tz1abs\companies...
05/09/2012	20:32:47.680	05/03/2012	4:46:13.210	F00000000	12	Excel	C:\dump\nxxtest\logs\20...
05/09/2012	20:32:47.680	05/03/2012	14:44:51.777	F00000000	13	Excel	C:\dump\nxxtest\logs\20...
05/09/2012	20:32:47.680	05/03/2012	14:05:52.193	F00000000	14	Excel	C:\dump\nxxtest\logs\20...
05/10/2012	17:41:07.054	05/03/2012	12:55:28.838	F00000000	3	Excel	E:\class_v2\win32_proj...

The raw data is UTF-16 in nature, and the timestamp information is prepended to the filename.



4.2.6 OpenWithList Key

- `ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\?*\.OpenWithList`

The *OpenWithList* artifact has a separate subkey for each extension of a file that was opened. Within each extension subkey, the list associates which application is used to open a file with that specific extension. Each of the items within an extension is ranked by a letter, which indicates the order of application execution.

Example: `cafae -hive user.win7.hive -open_with > out.txt`

Artifact: FileExts openwithlist
Registry key: NTUSER.DAT\Software\Microsoft\windows\CurrentVersion\Explorer\FileExts\?*\.openwithlist

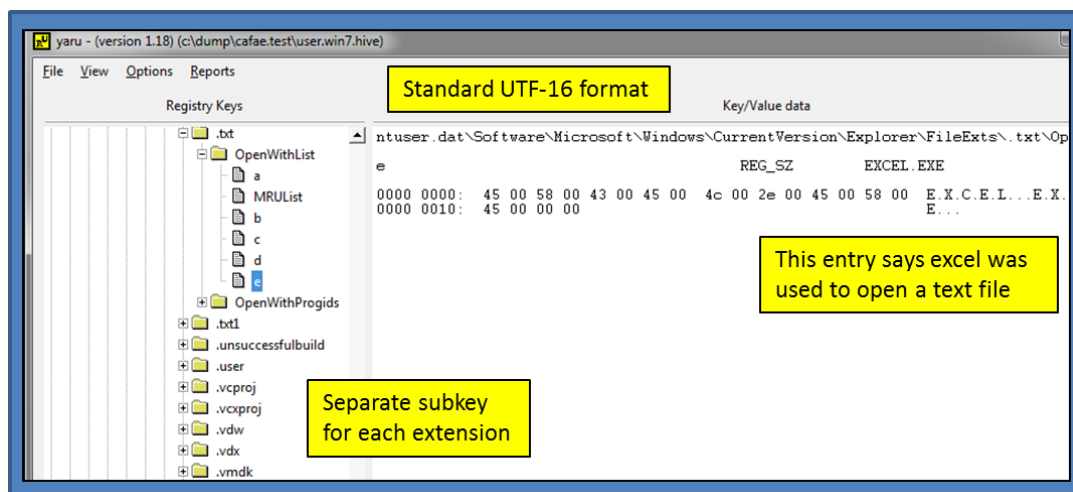
reg date	regtime[UTC]	item#	subkey	filename
05/10/2012	16:23:16.188	-> a	.txt	NOTEPAD.EXE
05/10/2012	16:23:16.188	b	.txt	WORDPAD.EXE
05/10/2012	16:23:16.188	c	.txt	EXCEL.EXE
05/10/2012	16:23:16.188	d	.txt	hworks64.exe
05/10/2012	16:23:16.188	e	.txt	vs.exe
05/08/2012	19:25:44.981	-> b	.xml	EXCEL.EXE
05/08/2012	19:25:44.981	a	.xml	MSOXMLB6.EXE
05/08/2012	17:49:08.389	-> a	.csv	EXCEL.EXE
05/08/2012	17:49:08.389	c	.csv	hworks64.exe
05/08/2012	17:49:08.389	d	.csv	vs.exe
05/08/2012	17:49:08.389	b	.csv	NOTEPAD.EXE
05/08/2012	17:01:39.655	-> a	.short	hworks64.exe
05/07/2012	12:55:27.892	-> a	.truncated	hworks64.exe
05/05/2012	16:23:57.071	-> a	.htm	ieplone.exe
05/05/2012	16:23:57.071	c	.htm	NOTEPAD.EXE
05/05/2012	16:23:57.071	b	.htm	WINWORD.EXE
04/30/2012	11:24:07.063	-> a	.cb	hworks64.exe
04/25/2012	11:48:38.211	-> a	.DS_Store	hworks64.exe
04/17/2012	14:52:48.450	-> a	.html	ieplone.exe
04/17/2012	14:52:48.450	b	.html	WINWORD.EXE

Grouped by extension

Within each extension sorted by MRU

Associated application that was used to open the file

The raw data is straight forward to parse since the data is UTF-16 in nature



4.2.7 ShellBag Keys

For completeness, this important set of registry keys is just mentioned here. This option, however, was not made available with *cafae*, since the *sbag* ^[2] tool was developed to specifically parse these registry artifacts.

4.3 Metadata Associated with Searching/Browsing

Previously, the options for *Search History*, *TypedURLs* and *Favorites* were broken out as separate options. Now they are combined into one option using the *-web* switch.

Example: *cafae -hive user.win7.hive -web > out.txt*

4.3.1 Search History

- *ntuser.dat\Software\Microsoft\Search Assistant\ACMru*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\WordWheelQuery*

For Windows XP, there is the *ACMru* key, which stores search terms that have been typed into a Windows search dialog box. The presence of subkeys, defined below, indicate where the search term was used:

- 5001 - List of terms used for the Internet Search Assistant
- 5603 - List of terms used for the Windows XP files and folders search
- 5604 - List of terms used in the "word or phrase in a file" search
- 5647 - List of terms used in the "for computers or people" search

Unfortunately, Vista did not include a registry key for user searches. Windows 7, however, defines the *WordWheelQuery* subkey to record information about user searches. Below is an example of the *WordWheelQuery* data.

Artifact: Search History - wordwheelquery
Registry key: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\wordwheelquery

reg date	regtime[UTC]	item#	filename
04/16/2012	12:43:03.320	-> 58	*.txt
04/16/2012	12:43:03.320	57	*.pot
04/16/2012	12:43:03.320	56	*.lrl
04/16/2012	12:43:03.320	55	usbdlm.h
04/16/2012	12:43:03.320	54	*.xls
04/16/2012	12:43:03.320	53	appverif
04/16/2012	12:43:03.320	51	*.tax
04/16/2012	12:43:03.320	50	vmwaretools
04/16/2012	12:43:03.320	49	mtdll.h
04/16/2012	12:43:03.320	48	dbgint.h
04/16/2012	12:43:03.320	47	*.bmp
04/16/2012	12:43:03.320	46	ddkwiz

Sorted byte MRUListEx entry

The WordWheelQuery data is straight forward to parse since it is in standard UTF-16 format.

yaru - (version 1.18) (c:\dump\cafae.test\user.win7.hive)

File View Options Reports

Registry Keys

WordWheelQuery

MRUListEx

0

2

3

4

6

7

Key/Value data

ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\WordWheelQuery\

6

REG_BINARY

0000 0000: 61 00 64 00 76 00 61 00 70 00 69 00 33 00 32 00 a.d.v.a.p.i.3.2

0000 0010: 2e 00 00 00

Standard UTF-16 format

4.3.2 TypedURLs Key

- ntuser.dat\Software\Microsoft\Internet Explorer\TypedURLs

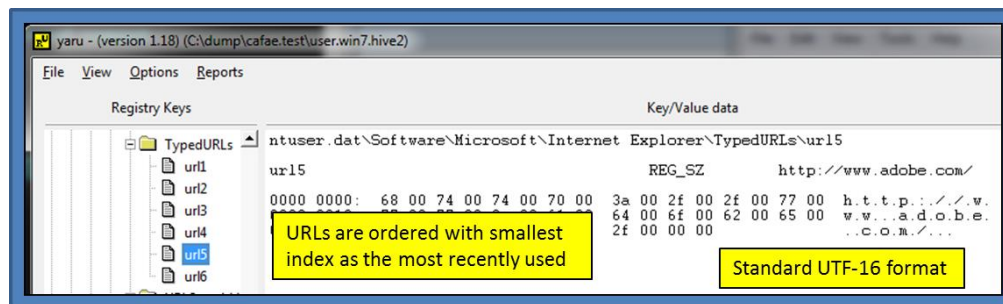
When a user types, or adds via a copy/paste, a URL directly into the browser, the *TypedURLs* subkey is updated. The list of URLs is sorted by number. The lowest number is the last, or most recently typed, URL.

Artifact: Typed URLs
Registry key: NTUSER.DAT\Software\Microsoft\Internet Explorer\TypedURLs

reg	regtime[UTC]	item#	path/filename
05/28/2012	18:53:35.535	-> 1	http://msdn.microsoft.com/
05/28/2012	18:53:35.535	2	http://www.amazon.com/
05/28/2012	18:53:35.535	3	http://www.google.com/
05/28/2012	18:53:35.535	4	http://www.yahoo.com/
05/28/2012	18:53:35.535	5	http://www.adobe.com/
05/28/2012	18:53:35.535	6	http://www.msn.com/

Entries sorted by index number

Every new URL will contain a separate entry and the contents of the URL are in UTF-16 format. The names are indexed based on the most recently used as the lowest index. Conversely, the highest index equates to the oldest entry.



4.3.3 Favorites Key

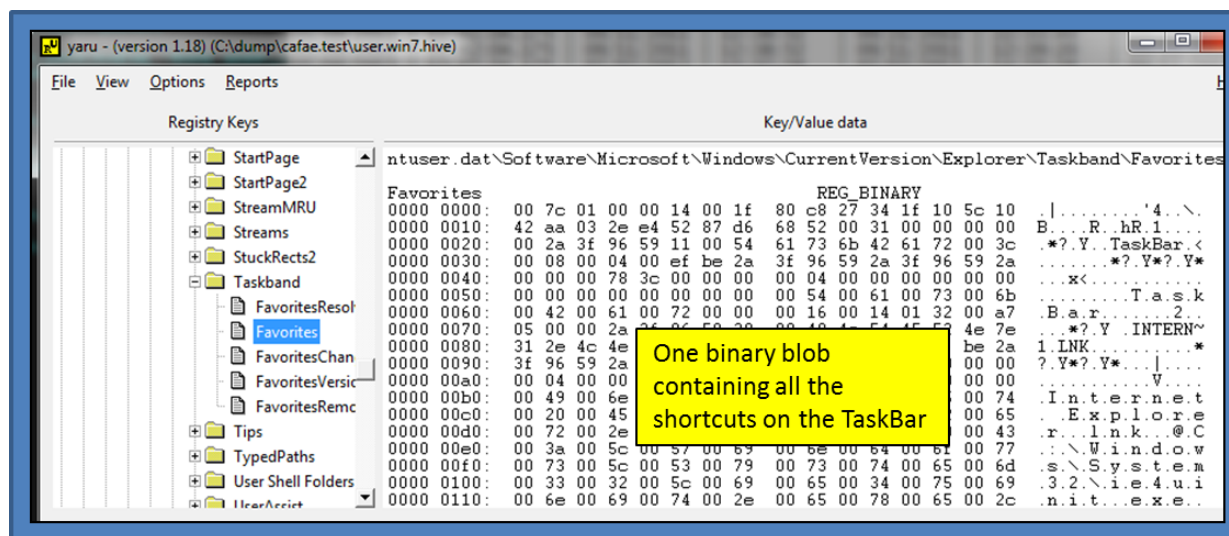
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage\Favorites*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\StartPage2\Favorites*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\Taskband\Favorites*

This set of keys covers the shortcuts on the *Windows Start Menu* and the *TaskBar*, respectively. Shown below, is an example of the parsed output of the *TaskBar* shortcuts. Some of the data in the figure is truncated to the right of the output. The truncated data includes extra metadata that was available that could be parsed out.

Artifact: Taskband Favorites
Registry key: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\Taskband\Favorites

reg date	regtime[UTC]	mdate	time [UTC]	adate	program/link
05/09/2012	12:42:04.175	09/10/2011	11:12:44	09/10/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Internet Explorer...
05/09/2012	12:42:04.175	07/14/2009	04:49:40	09/10/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Windows Explorer.1
05/09/2012	12:42:04.175	11/21/2010	03:40:32	09/10/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Windows Media Play
05/09/2012	12:42:04.175	09/11/2011	12:05:16	09/11/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Microsoft Excel 20
05/09/2012	12:42:04.175	09/11/2011	12:05:16	09/11/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Microsoft PowerPo
05/09/2012	12:42:04.175	09/11/2011	12:05:16	09/11/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Microsoft Word 201
05/09/2012	12:42:04.175	09/11/2011	14:09:24	09/11/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Microsoft Visio 20
05/09/2012	12:42:04.175	10/28/2011	22:11:26	11/08/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Command Prompt.lnk
05/09/2012	12:42:04.175	09/11/2011	12:50:02	09/11/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\GhostCast Server.1
05/09/2012	12:42:04.175	10/23/2011	17:24:42	10/23/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\WinDbg.lnk
05/09/2012	12:42:04.175	11/18/2012	19:49:24	01/18/2013	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\VMware Workstation
05/09/2012	12:42:04.175	09/11/2011	01:28:54	09/11/2011	{1f3427c8-5c10-4210-aa03-2ee45287d668}\TaskBar\Microsoft Visual S

The raw data of the *TaskBar* is shown below. The binary data is one blob containing all the shortcuts and related metadata.



4.4 Network Related Artifacts found in User Hives

- *ntuser.dat\Software\Microsoft\Terminal Server Client\Servers*
- *ntuser.dat\Software\Microsoft\Terminal Server Client\Default\AddIns\RDPDR*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\Map Network Drive MRU*
- *ntuser.dat\Network\Z*
- *ntuser.dat\Network\Software\Martin Prikryl\WinSCP 2\Sessions*
- *ntuser.dat\Network\Software\SimonTatham\PuTTY\Sessions*

Example: *cafae -hive user.win7.hive -network > out.txt*

4.5 Volume Related Artifacts found in User Hives

- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2*

Example: *cafae -hive user.win7.hive -volumes > out.txt*

4.6 Computer Metadata Related Artifacts found in User Hives

This last subsection is just a catch-all for other useful artifacts that pertain to the computer configuration as set by, or indirectly affected by, the user. Some of the registry keys include:

- *ntuser.dat\Environment*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\Desktop*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ControlPanel*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket\LastEnum*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket\Volume\?*MaxCapacity*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\Wallpaper\MRU*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\Windows*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\Logon User Name*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\Winlogon*
- *ntuser.dat\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Explorer\ComputerDescriptions*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Compatibility Assistant*
- *ntuser.dat\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Layers*
- *ntuser.dat\Control Panel\don't load*
- *ntuser.dat\Software\Microsoft\Windows\CurrentVersion\Internet Settings*

Example: *cafae -hive user.win7.hive -computer > out.txt*

Artifact: MountPoints2				
Registry key: NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2				
reg date	regtime[UTC]	regpath		
05/10/2012	18:05:36.209	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{b35507d4-dc99-11e0-95ba-005056c00008}		
05/10/2012	12:15:51.096	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{6f196eff-6c3d-11e1-8eef-005056c00008}		
04/29/2012	00:32:41.325	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{95847cd6-4e66-11e1-8640-005056c00008}		
04/29/2012	00:27:47.473	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{52c1634e-f419-11e0-89f5-005056c00008}		
04/29/2012	00:26:57.707	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{9735effc-dbb6-11e0-beb1-998380bd2744}		
04/29/2012	00:26:48.209	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{8afc4062-9183-11e1-a64c-005056c00008}		
04/20/2012	17:08:14.953	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{48574cdb-87c1-11e1-860c-005056c00008}		
04/16/2012	19:32:24.034	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{1d913597-87ed-11e1-8fd8-005056c00008}		
04/16/2012	19:28:59.787	NTUSER.DAT\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\{1d913597-87ed-11e1-8fd8-005056c00008}		
Artifact: Windows				
Registry key: NTUSER.DAT\Software\Microsoft\Windows NT\CurrentVersion\Windows				
reg date	regtime[UTC]	value name	value data	
10/02/2011	13:33:50.532	UserSelectedDefault	1	
10/02/2011	13:33:50.532	Device	Canon MF4500 Series UFRII LT,winspool,Ne03:	
Artifact: PrinterPorts				
Registry key: NTUSER.DAT\Software\Microsoft\Windows NT\CurrentVersion\PrinterPorts				
reg date	regtime[UTC]	value name	value data	
05/10/2012	12:15:35.765	Microsoft XPS Document Writer	winspool,Ne01:,15,45	
05/10/2012	12:15:35.765	Fax	winspool,Ne02:,15,45	
05/10/2012	12:15:35.765	Send To OneNote 2010	winspool,null:,15,45	
05/10/2012	12:15:35.765	Snagit 10	winspool,Ne00:,15,45	
05/10/2012	12:15:35.765	Samsung ML-3050 Series (Copy 1)	winspool,lp,15,45	
05/10/2012	12:15:35.765	Canon MF4500 Series UFRII LT	winspool,Ne03:,15,45	

4.7 Persistence Related Artifacts found in User Hives

This includes a combination of a number of previous keys used for other options as well as some new ones collected in one place to pull persistence type data.

Example: *cafae -hive user.win7.hive -persistence > out.txt*

5 System Hive Artifacts

5.1 Timezone

- HKLM\SYSTEM\CurrentControlSet\Control\TimeZoneInformation

Example: *cafae -hive system_hive -timezone > out.txt*


```
cmdline: cafae64 -hive c:\dump\System -computer
```

Artifact: Timezone Info
Registry key: HKLM\System\CurrentControlSet\Control\TimeZoneInformation [enumvalues]

reg date	reg-UTC	value name	value data	In minutes
04/09/2014	17:32:30.000	Bias	0x0000012c (300)	
04/09/2014	17:32:30.000	DaylightBias	0xffffffffc4 (-60)	
04/09/2014	17:32:30.000	DaylightName	@tzres.dll,-111	
04/09/2014	17:32:30.000	StandardBias	0x00000000 (0)	
04/09/2014	17:32:30.000	StandardName	@tzres.dll,-112	
04/09/2014	17:32:30.000	DynamicDaylightTimeDisabled	0x00000000 (0)	
04/09/2014	17:32:30.000	ActiveTimeBias	0x000000f0 (240)	
04/09/2014	17:32:30.000	DaylightStart	00 00 03 00 02 00 02 00 00	
04/09/2014	17:32:30.000	StandardStart	00 00 0b 00 01 00 02 00 00	
04/09/2014	17:32:30.000	TimeZoneKeyName	Eastern Standard Time	

To use the above information, one can use the following relationships:

Local Time = UTC – ActiveTimeBias

Standard Time = Bias + StandardBias

Daylight Time = Bias + DaylightBias

5.2 Devices

- HKLM\SYSTEM\CurrentControlSet\Enum

Example: *cafae -hive system_hive -devices > out.txt*

```
cmdline: cafae64 -hive c:\dump\win8.dblake.SYSTEM -devices
```

Artifact: EnumDevices
Registry key: HKLM\System\CurrentControlSet\Enum\?*\\?* [targeting certain values]

reg date	reg-UTC	subkey	Service	FriendlyName	DeviceDesc
...					
09/23/2013	19:14:45.985	BTH\ms_bthbrb\8836e..	BthEnum		@bth.inf,...
09/23/2013	19:14:45.532	BTH\ms_bthle\8836e5..	BthLEEnum		@bthleenum...
09/23/2013	19:14:46.173	BTH\ms_bthpan\8836e..	BthPan	Bluetooth Device (Pers...	@bthpan.in...
09/23/2013	19:14:45.767	BTH\ms_rfcomm\8836e..	RFCOMM	Bluetooth Device (RFCO...	@tdibth.in...
09/23/2013	19:13:58.317	DISPLAY\auo205c\188..	monitor		@monitor.3...
09/23/2013	19:14:24.215	DISPLAY\auo205c\482..	monitor		@monitor.3...
09/23/2013	19:14:31.192	HDAUDIO\func_01&ven..	CnxtHdAudSer		@oem44.in...
10/13/2013	10:31:26.836	HDAUDIO\func_01&ven..	IntcDAud		@oem35.in...
09/23/2013	19:14:29.418	HID\vid_03eb&pid_88..			@input.in...
09/23/2013	19:14:29.418	HID\vid_03eb&pid_88..			@input.in...
...					
10/02/2013	21:15:10.189	SWD\dafupnpprovider..		VALHALLA: dblake@asgar..	@c_swdevic...
10/21/2013	19:51:51.577	SWD\dafupnpprovider..		VALHALLA: dblake@asgar..	@c_swdevic...
09/23/2013	19:51:32.997	SWD\mmdevapi\{0.0.0..		Digital Output (2- Int...	@AudioEndp...
10/19/2013	02:06:40.184	SWD\mmdevapi\{0.0.0..		Digital Output (2- Int...	@AudioEndp...
09/23/2013	19:14:31.536	SWD\mmdevapi\{0.0.0..		Speakers (Conexant Sma...	@AudioEndp...
10/22/2013	16:23:34.189	SWD\mmdevapi\{3.0.0..		Remote Audio	@AudioEndp...
09/23/2013	19:14:42.338	SWD\msdas\{ce958e9a..		Microsoft Device Assoc...	@c_swdevic...
09/23/2013	19:24:31.901	SWD\printenum\{0acc..		Canon MX870 series Pri...	@PrintQueu...
09/23/2013	20:49:30.625	SWD\printenum\{5449..		Send To OneNote 2013 ..	@PrintQueu...
10/04/2013	17:44:43.727	SWD\printenum\{5d6b..		\\VALHALLA\Canon MX870...	@PrintQueu...
09/23/2013	19:24:38.700	SWD\printenum\{5de2..		Canon MX870 series FAX...	@PrintQueu...
09/23/2013	19:23:36.390	SWD\printenum\{d943..		Microsoft XPS Document...	@PrintQueu...
10/17/2013	19:28:34.742	SWD\wpdbusenum\?_?..	WUDFWpdFs	PHOTOS BACK	Drive SM...
10/22/2013	21:41:54.248	SWD\wpdbusenum\?_?..	WUDFWpdFs	SUBJECT	TACTICAL...
10/18/2013	18:33:26.141	SWD\wpdbusenum\?_?..	WUDFWpdFs	FILES	Moser Baer...
10/13/2013	09:03:48.350	SWD\wpdbusenum\?_?..	WUDFWpdFs	RECOVERY	Card Read...
10/21/2013	20:11:48.558	SWD\wpdbusenum\?_?..	WUDFWpdFs	F:\	USB DISK...
10/17/2013	21:06:16.522	SWD\wpdbusenum\?_?..	WUDFWpdFs	BLAKE FILES	Flash Disk...
10/18/2013	18:32:19.506	SWD\wpdbusenum\{8df..	WUDFWpdFs	My Passport	1600BEV E...
10/23/2013	03:09:14.768	SWD\wpdbusenum\{abc..	WUDFWpdFs	ACQUISITION	2500BMV E...
...					
10/22/2013	16:23:28.429	UMB\umb\1&841921d&0..	umbus		@umbus.in...
09/23/2013	19:14:29.293	USBSTOR\cdrom&ven_h..	cdrom	HL-DT-ST DVDROM GP08NU...	@cdrom.in...
10/17/2013	19:28:33.442	USBSTOR\disk&ven_f1..	disk	Flash Drive SM_USB20 U...	@disk.inf...
10/22/2013	21:41:53.373	USBSTOR\disk&ven_fr..	disk	FRESPONS TACTICAL_Subj...	@disk.inf...
10/18/2013	18:33:24.482	USBSTOR\disk&ven_mb..	disk	MBIL SSM Moser Baer Di...	@disk.inf...
10/13/2013	09:03:25.259	USBSTOR\disk&ven_mu..	disk	Multiple Card Reader ..	@disk.inf...
10/21/2013	18:46:16.382	USBSTOR\disk&ven_sm..	disk	SMI USB DISK USB Devic...	@disk.inf...
10/17/2013	21:06:15.791	USBSTOR\disk&ven_us..	disk	USB2.0 Flash Disk USB ..	@disk.inf...
10/18/2013	18:32:18.738	USBSTOR\disk&ven_wd..	disk	WD 1600BEV External US...	@disk.inf...
10/23/2013	03:09:13.933	USBSTOR\disk&ven_wd..	disk	WD 2500BMV External US...	@disk.inf...
10/21/2013	20:40:25.385	USB\vid_0421&pid_06..	WUDFWpdMtp	Donald's Windows Phone...	Lumia 928...
10/19/2013	19:40:14.943	USB\vid_0421&pid_06..	WUDFWpdMtp	RM-860/Nokia Lumia 928	@vntush.i...

5.3 Shimcache

- *HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility\AppCompatCache*
 - Used for WinXP
- *HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache\AppCompatCache*
 - Used for Win2003 and beyond

Example: *cafae -hive system_hive -shimcache > out.txt*

The **-shimcache** option looks at the CurrentControlSet. To explicitly look at a specific ControlSet (eg. ControlSet001, ControlSet002, ..), one could use the following command:

cafae -hive system_hive -key "HKLM\System\ControlSet001\Control\Session Manager\AppCompatCache\AppCompatCache" -arg "-shim_cache"

The **-key** option tells **cafae** which key to target and more specifically which ControlSet00x to look at. The **-arg “-shim_cache”** is tells **cafae** to use one of the internal script engine commands to parse the value data at the key as if it were in a shim cache format.

5.4 Computer Related Artifacts found in System Hives

- *HKLM\System\CurrentControlSet\Control\TimeZoneInformation*
- *HKLM\System\CurrentControlSet\Control\Windows*
- *HKLM\System\CurrentControlSet\Control\CrashControl*
- *HKLM\System\CurrentControlSet\Control*
- *HKLM\System\CurrentControlSet\Control\Watchdog\Display*
- *HKLM\System\CurrentControlSet\Control\FileSystem*
- *HKLM\System\CurrentControlSet\Control\ComputerName\ComputerName*
- *HKLM\System\CurrentControlSet\Control\Session Manager\Memory Management*
- *HKLM\System\CurrentControlSet\Control\SecurityProviders*
- *HKLM\System\CurrentControlSet\Control\Lsa*

Example: **cafae -hive system_hive -computer > out.txt**

```
cmdline: c:\fae64 -hive c:\dump\win8.d\blake.SYSTEM -computer
```

Artifact: Windows Version
Registry key: HKLM\System\CurrentControlSet\Control\Windows [enumvalues]

reg date	reg-UTC	value name	value data
10/23/2013	02:56:04.383	Directory	%SystemRoot%
10/23/2013	02:56:04.383	ShutdownTime	10/23/13 02:56:04.383
10/23/2013	02:56:04.383	SystemDirectory	C:\WINDOWS\SysWOW64
...			

Artifact: CrashControl
Registry key: HKLM\System\CurrentControlSet\Control\CrashControl [enumvalue]

reg date	reg-UTC	value name	value data
09/23/2013	19:14:08.558	AutoReboot	0x00000001 (1)
09/23/2013	19:14:08.558	CrashDumpEnabled	0x00000007 (7)
09/23/2013	19:14:08.558	DumpFile	%SystemRoot%\MEMORY.DMP
09/23/2013	19:14:08.558	MinidumpDir	%SystemRoot%\Minidump
09/23/2013	19:14:08.558	MinidumpsCount	0x00000032 (50)
09/23/2013	19:14:08.558	Overwrite	0x00000001 (1)
...			

Artifact: FileSystem Flags
Registry key: HKLM\System\CurrentControlSet\Control\FileSystem [enumvalues]

Access timestamps are disabled

reg date	reg-UTC	value name	value data
09/23/2013	19:14:08.573	NtfsDisable8dot3NameCreation	0x00000002 (2)
09/23/2013	19:14:08.573	NtfsDisableLastAccessUpdate	0x00000001 (1)
09/23/2013	19:14:08.573	RefsDisableLastAccessUpdate	0x00000001 (1)
...			

Artifact: Computer Name
Registry key: HKLM\System\CurrentControlSet\Control\ComputerName\ComputerName

reg date	reg-UTC	value name	value data
09/23/2013	19:14:08.558	ComputerName	BIFROST

Artifact: Memory Management
Registry key: HKLM\System\CurrentControlSet\Control\Session Manager\Memory

reg date	reg-UTC	value name	value data
10/23/2013	10:56:24.429	ClearPageFileAtShutdown	0x00000000 (0)
10/23/2013	10:56:24.429	DisablePagingExecutive	0x00000000 (0)
10/23/2013	10:56:24.429	PagingFiles	?:\pagefile.sys

5.5 Network Related Artifacts found in System Hives

- HKLM\System\CurrentControlSet\Control\Network\{4D36E972-E325-11CE-BFC1-08002BE10318}\?*\Connection
- HKLM\System\CurrentControlSet\Control\Network\{4d36e973-e325-11ce-bfc1-08002be10318}
- HKLM\System\CurrentControlSet\Control\Network\{4d36e974-e325-11ce-bfc1-08002be10318}
- HKLM\System\CurrentControlSet\Control\Network\{4d36e975-e325-11ce-bfc1-08002be10318}
- HKLM\System\CurrentControlSet\Services\Tcpip\Parameters\Interfaces
- HKLM\System\CurrentControlSet\Control\Terminal Server
- HKLM\System\CurrentControlSet\Control\Terminal Server\WinStations
- HKLM\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp
- HKLM\System\CurrentControlSet\Services\TermService\Parameters
- HKLM\System\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy

- HKLM\System\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy\FirewallRules
- HKLM\System\CurrentControlSet\Control\Terminal Server\Wds\rdpwd
- HKLM\System\CurrentControlSet\Services\LanmanServer\Shares
- HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Linkage
- HKLM\System\CurrentControlSet\Control\NetworkSetup2\Interfaces\{373CA98A-0E76-4A8D-97CD-9AD7DCB8C8D8}\Kernel

Example: *cafae -hive system_hive -network > out.txt*

```
cmdline: cafae64 -hive c:\dump\win8.dblake.SYSTEM -network
```

Artifact: LanmanServer
Registry key: HKLM\System\CurrentControlSet\Services\LanmanServer\Shares [enumvalues]

reg date	reg-UTC	value name	value data
10/13/2013	10:40:16.054	SkyDrive	CATimeout=0; CSCFlags=0; MaxUses=4294967295; Path=C:\Users\Donald\SkyDrive; Permissions=9; ShareName=S
10/13/2013	10:40:16.054	Users	CATimeout=0; CSCFlags=2048; MaxUses=4294967295; Path=C:\Users; Permissions=0; Remark=; ShareName=Users

Artifact: Network Service
Registry key: HKLM\System\CurrentControlSet\Control\Network\{4d36e974-e325-11ce-bfc1-08002be10318}\?* [targeting certain values]

reg date	reg-UTC	subkey	Description	ComponentId	InstallTimeStamp
09/23/2013	19:14:45.110	{5C8F81..4E3698}	Virtual WiFi Filter Driver	ms_vwifi	.. [09/23/2013 19:14:45.110]
09/23/2013	19:14:08.573	{EA24CD..8E83DD}	Microsoft NDIS Capture	MS_NDISCAP	.. [08/22/2013 14:45:59.165]
09/23/2013	19:14:08.573	{E7C3B2..D72E7A}	Microsoft Windows Filtering Platform	MS_WfpLwf_vSwitch	.. [08/22/2013 14:45:58.822]
09/23/2013	19:14:08.573	{E475CF..0E18A1}	NativeWiFi Filter	MS_NativeWifiP	.. [08/22/2013 14:45:55.494]

Artifact: Network Transport
Registry key: HKLM\System\CurrentControlSet\Control\Network\{4d36e975-e325-11ce-bfc1-08002be10318}\?* [targeting certain values]

reg date	reg-UTC	subkey	Description	ComponentId	InstallTimeStamp
09/23/2013	19:14:08.589	{F4C870..2F0D83D}	Microsoft TCP/IP version 6 - Tunnels	MS_TCPIP6_TUNNEL	.. [08/22/2013 14:45:56.353]
09/23/2013	19:14:08.589	{F0A0A4..52D76EB}	SSTP based VPN	ms_sstp	.. [08/22/2013 14:45:52.603]
09/23/2013	19:14:08.589	{E8B167..7CD2F66}	Layer 2 Tunneling Protocol	ms_l2tp	.. [08/22/2013 14:45:52.415]
09/23/2013	19:14:08.589	{E2BC73..E485A1C}	Link-Layer Topology Discovery Responder	MS_RSPNDR	.. [08/22/2013 14:45:54.947]

Artifact: DHCP Details
Registry key: HKLM\System\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\?* [targeting certain values]

reg date	reg-UTC	subkey	DhcpDomain	DhcpIPAddress	DhcpServer	LeaseObtainedTime	LeaseTermi
10/23/2013	02:56:23.062	{518549..F1A64}	talonne	192.168.1.49	192.168.1.1	.. [10/23/2013 02:56:23.000]	.. [10/24/
10/21/2013	19:34:41.162	{518549..F1A64}\C4F4..	key.chilli..	192.168.182.95	192.168.182.1	.. [10/21/2013 19:24:46.000]	.. [10/21/
10/14/2013	06:26:26.132	{518549..F1A64}\3616..	prg.aero	10.100.65.74	10.100.65.3	.. [10/14/2013 06:26:12.000]	.. [10/14/
10/14/2013	06:26:06.617	{518549..F1A64}\0727..	prg.aero	10.17.14.218	10.17.0.7	.. [10/14/2013 06:24:08.000]	.. [10/14/
10/01/2013	14:18:29.051	{518549..F1A64}\3516..	home	192.168.1.10	192.168.1.1	.. [10/01/2013 14:18:23.000]	.. [10/02/
09/26/2013	18:15:57.065	{518549..F1A64}\4496..		192.168.1.102	192.168.1.1	.. [09/26/2013 17:51:05.000]	.. [09/27/

5.6 System Restore Related Artifacts found in System Hives

- HKLM\SYSTEM\CurrentControlSet\Control\BackupRestore\FilesNotToSnapshot
- HKLM\SYSTEM\CurrentControlSet\Control\BackupRestore\FilesNotToBackup
- HKLM\SYSTEM\CurrentControlSet\Control\BackupRestore\KeysNotToRestore

Example: *cafae -hive system_hive -restore > out.txt*

5.7 Services Related Artifacts found in System Hives

The services are sorted by Start type. Any service without a start value is not shown in the output.

- *HKLM\System\CurrentControlSet\Services*

Example: *cafae -hive system_hive -services > out.txt*

```
cmdline: cafae64 -hive c:\dump\win8.d\lake.SYSTEM -services
```

Artifact: Services with start						
Registry key: HKLM\System\CurrentControlSet\Services [targeting certain values]						

reg date	reg-UTC	subkey	Start	Type	Group	ImagePath
09/23/2013	19:14:08.667	Services\3ware	0x00000000 (0)	0x00000001 (1)	SCSI miniport	System32\drivers\3ware.sys
10/23/2013	02:56:11.058	Services\ACPI	0x00000000 (0)	0x00000001 (1)	Core	System32\drivers\ACPI.sys
09/23/2013	19:14:08.667	Services\ADP80XX	0x00000000 (0)	0x00000001 (1)	SCSI Miniport	System32\drivers\ADP80XX.SYS
09/23/2013	19:14:08.683	Services\CLFS	0x00000000 (0)	0x00000001 (1)	Filter	System32\drivers\CLFS.sys
09/23/2013	19:14:08.683	Services\CNG	0x00000000 (0)	0x00000001 (1)	Core	System32\Drivers\cng.sys
10/23/2013	02:56:11.308	Services\EhStorClass	0x00000000 (0)	0x00000001 (1)	SCSI Class	System32\drivers\EhStorClass.sys
09/23/2013	19:14:08.683	Services\EhStorTcgDrv	0x00000000 (0)	0x00000001 (1)	SCSI Class	System32\drivers\EhStorTcgDrv.sys
09/23/2013	19:14:08.714	Services\FileInfo	0x00000000 (0)	0x00000002 (2)	FSFilter Bottom	System32\drivers\fileinfo.sys
09/23/2013	19:14:08.714	Services\FltMgr	0x00000000 (0)	0x00000002 (2)	FSFilter Infrast..	system32\drivers\fltmgr.sys
09/23/2013	19:14:08.714	Services\Fs_Rec	0x00000000 (0)	0x00000008 (8)	File System	
09/23/2013	19:14:08.714	Services\HpSAMD	0x00000000 (0)	0x00000001 (1)	SCSI Miniport	System32\drivers\HpSAMD.sys
09/23/2013	19:14:08.730	Services\KSecDD	0x00000000 (0)	0x00000001 (1)	Base	System32\Drivers\ksecdd.sys
09/23/2013	19:14:08.730	Services\KSecPkg	0x00000000 (0)	0x00000001 (1)	Cryptography	System32\Drivers\ksecpkg.sys
09/23/2013	19:14:08.730	Services\LSI_SAS	0x00000000 (0)	0x00000001 (1)	SCSI Miniport	System32\drivers\lsi_sas.sys
09/23/2013	19:14:08.730	Services\LSI_SAS2	0x00000000 (0)	0x00000001 (1)	SCSI Miniport	System32\drivers\lsi_sas2.sys
...						

5.8 Persistence Related Artifacts found in System Hives

- *HKLM\System\CurrentControlSet\Control*
- *HKLM\System\CurrentControlSet\Control\BootVerificationProgram*
- *HKLM\System\CurrentControlSet\Control\Control\Control\Session Manager*
- *HKLM\System\CurrentControlSet\Control\Lsa*
- *HKLM\System\CurrentControlSet\Control\NetworkProvider\Order*
- *HKLM\System\CurrentControlSet\Control\Print\Monitors*
- *HKLM\System\CurrentControlSet\Control\SafeBoot*
- *HKLM\System\CurrentControlSet\Control\SecurityProviders*
- *HKLM\System\CurrentControlSet\Control\Session Manager*
- *HKLM\System\CurrentControlSet\Control\Session Manager\KnownDlls*
- *HKLM\System\CurrentControlSet\Control\Terminal Server\Wds\rdpwd*
- *HKLM\System\CurrentControlSet\Services*
- *HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\Protocol_Catalog5*
- *HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\Protocol_Catalog9*

Example: *cafae -hive system_hive -persistence > out.txt*

5.9 Background/Desktop Activity Moderator

- `HKLM\System\CurrentControlSet\Services\bam\UserSettings`
- `HKLM\System\CurrentControlSet\Services\bam\State\UserSettings`
- `HKLM\System\CurrentControlSet\Services\dam\UserSettings`
- `HKLM\System\CurrentControlSet\Services\dam\State\UserSettings`

Example: `cafae -hive system_hive -bam > out.txt`

5.10 All System artifacts

One can parse artifacts from a system hive using the `-all_system` switch.

6 Software Hive Artifacts

6.1 Operating System

- *HKLM\Software\Microsoft\Windows NT\CurrentVersion*
- *HKLM\Software\Microsoft\Windows NT\CurrentVersion\ProfileList*

Example: *cafae -hive software_hive -computer > out.txt*

cmdline: cafae64 -hive c:\SOFTWARE -computer			

Artifact: Operating System Install Date			
Registry key: HKLM\Software\Microsoft\Windows NT\CurrentVersion [targeting certain values]			

reg date	reg-UTC	subkey	InstallDate
05/14/2014	14:49:24.345	CurrentVersion	0x51116295 [02/05/2013 19:50:45.000]

Artifact: Operating System			
Registry key: HKLM\Software\Microsoft\Windows NT\CurrentVersion [enumvalues]			

reg date	reg-UTC	value name	value data
05/14/2014	14:49:24.345	CurrentVersion	6.1
05/14/2014	14:49:24.345	CurrentBuild	7601
05/14/2014	14:49:24.345	SoftwareType	System
05/14/2014	14:49:24.345	CurrentType	Multiprocessor Free
05/14/2014	14:49:24.345	InstallDate	0x51116295 (1360093845)
05/14/2014	14:49:24.345	RegisteredOrganization	
05/14/2014	14:49:24.345	RegisteredOwner	test
05/14/2014	14:49:24.345	SystemRoot	C:\Windows
05/14/2014	14:49:24.345	InstallationType	Client
05/14/2014	14:49:24.345	EditionID	Professional
05/14/2014	14:49:24.345	ProductName	Windows 7 Professional
05/14/2014	14:49:24.345	ProductId	00371-220-8503041-86684
05/14/2014	14:49:24.345	DigitalProductId	a4 00 00 00 03 00 00 00 30 30 33 37 31 2d 3
05/14/2014	14:49:24.345	DigitalProductId4	f8 04 00 00 04 00 00 00 30 00 30 00 33 00 3
05/14/2014	14:49:24.345	CurrentBuildNumber	7601
05/14/2014	14:49:24.345	BuildLab	7601.win7sp1_gdr.140303-2144
05/14/2014	14:49:24.345	BuildLabEx	7601.18409.amd64fre.win7sp1_gdr.140303-2144
05/14/2014	14:49:24.345	BuildGUID	9f922bc2-f1e4-4791-be6f-4ccda0d35f6f
05/14/2014	14:49:24.345	CSDBuildNumber	1130
05/14/2014	14:49:24.345	PathName	C:\Windows
05/14/2014	14:49:24.345	CSDVersion	Service Pack 1

Artifact: ProfileList			
Registry key: HKLM\Software\Microsoft\Windows NT\CurrentVersion\ProfileList\?* [targeting certain values]			

reg date	reg-UTC	subkey	ProfileImagePath
07/14/2009	04:53:25.780	S-1-5-18	%systemroot%\system32\winlogon\profiles\default\userprofile
02/05/2013	22:46:03.602	S-1-5-19	C:\Windows\ServiceProfiles\LocalService
02/05/2013	22:46:03.602	S-1-5-20	C:\Windows\ServiceProfiles\NetworkService
07/28/2014	17:14:42.744	S-1-5-21-1684986164-3766221403-2002402928-1000	C:\Users\testuser
02/11/2014	19:49:44.176	S-1-5-21-1684986164-3766221403-2002402928-1002	C:\Users\dummyuser

6.2 Class Identifiers (CLSIDs)

- *HKLM\Software\Classes\CLSID*
- *HKLM\Software\Wow6432Node\Classes\CLSID*

Example: *cafae -hive software_hive -clsid > out.txt*

6.3 In Process Servers (InProcServers)

- HKLM\Software\Classes\CLSID\?*InProcServer32
- HKLM\Software\Wow6432Node\Classes\CLSID\?*InProcServer32

Example: *cafae -hive software_hive -inprocservers > out.txt*

6.4 Codecs

- HKLM\Software\Microsoft\Windows NT\CurrentVersion\Drivers32
- HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Drivers32
- HKLM\Software\Classes\Filter
- HKLM\Software\Classes\CLSID\?*Instance
- HKLM\Software\Wow6432Node\Classes\CLSID\?*Instance

Example: *cafae -hive software_hive -codecs > out.txt*

6.5 Desktop related keys (Explorer)

- HKLM\Software\Microsoft\Driver Signing
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\AeDebug
- HKLM\Software\Microsoft\RemovalTools\MRT
- HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\BitBucket
- HKLM\Software\Classes\Protocols
- HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\ShellExecuteHooks
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Explorer\ShellExecuteHooks
- HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\ShellIconOverlayIdentifiers
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Explorer\ShellIconOverlayIdentifiers
- HKLM\Software\Classes\?*ShellEx\ContextMenuHandlers
- HKLM\Software\Wow6432Node\Classes\?*ShellEx\ContextMenuHandlers
- HKLM\Software\Classes\?*ShellEx\DragDropHandlers
- HKLM\Software\Wow6432Node\Classes\?*ShellEx\DragDropHandlers
- HKLM\Software\Classes\?*ShellEx\PropertySheetHandlers
- HKLM\Software\Wow6432Node\Classes\?*ShellEx\PropertySheetHandlers
- HKLM\Software\Classes\?*ShellEx\CopyHookHandlers
- HKLM\Software\Wow6432Node\Classes\?*ShellEx\CopyHookHandlers
- HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensions\Approved
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Shell Extensions\Approved
- HKLM\Software\Microsoft\Windows\CurrentVersion\ShellServiceObjectDelayLoad
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\ShellServiceObjectDelayLoad
- HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\AutoplayHandlers\Handlers
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
- HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Image File Execution Options

Example: *cafae -hive software_hive -explorer > out.txt*

6.6 Installed Software

- *HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall*
- *HKLM\Software\Microsoft\Windows\CurrentVersion\App Paths*
- *HKLM\Software\Microsoft\Active Setup\Installed Components*
- *HKLM\Software\Wow6432Node\Microsoft\Active Setup\Installed Components*
- *HKLM\Software\Classes\Installer\Products*
- *HKLM\Software\Microsoft\Windows\CurrentVersion\Installer\UserData*\Products*

Example: *cafae -hive software_hive -installed_sw > out.txt*

6.7 EmdMgmt

- *HKLM\Software\Microsoft\Windows NT\CurrentVersion\EMDMgmt*

Example: *cafae -hive software_hive -emdmgmt > out.txt*

6.8 Shell Spawning of an Application

- *HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows*
- *HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows*
- *HKLM\Software\Microsoft\Windows NT\CurrentVersion\Accessibility*
- *HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Accessibility*
- *HKLM\Software\Classes\Folder\shellex\ColumnHandlers*
- *HKLM\Software\Classes*\shell*\command*

Example: *cafae -hive software_hive -emdmgmt > out.txt*

6.9 Run key Related Artifacts found in Software Hives

- *HKLM\Software\Microsoft\Windows\CurrentVersion\Run*
- *HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Run*
- *HKLM\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce*
- *HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\RunServicesOnce*
- *HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices*
- *HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\RunServices*
- *HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce*
- *HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\RunOnce*
- *HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx*
- *HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\RunOnceEx*
- *HKLM\Software\Microsoft\Windows NT\Terminal Server\Install\Software\Microsoft\Windows\CurrentVersion\Run*
- *HKLM\Software\Microsoft\Windows NT\Terminal Server\Install\Software\Microsoft\Windows\CurrentVersion\Runonce*

- HKLM\Software\Microsoft\Windows NT\Terminal Server\Install\Software\Microsoft\Windows\CurrentVersion\RunonceEx
- HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
- HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
- HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon

Example: *cafae -hive software_hive -runkeys > out.txt*

6.10 Network Related Artifacts found in Software Hives

- HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkCards
- HKLM\Software\Microsoft\WZCSVC\Parameters\Interfaces
- HKLM\Software\Microsoft\Windows\CurrentVersion\HomeGroup\NetworkLocations
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkList\Profiles
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkList\Signatures
- HKLM\Software\Microsoft\MSSQLServer\Client\SuperSocketNetLib\LastConnect

Example: *cafae -hive software_hive -network > out.txt*

```
cmdline: cafae64 -hive c:\dump\win8.d\blake.software -network
```

Artifact: NetworkCards

Registry key: HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkCards\?* [enumvalues]

reg date	reg-UTC	subkey	value name	value data
09/23/2013	19:14:44.876	1	Description	Realtek RTL8723A Wireless LAN 802.11n USB 2.0 Network Adapter
09/23/2013	19:14:44.876	1	ServiceName	{5185491C-401D-491E-8C6F-07F6AFFF1A64}

Artifact: Network Profiles

Registry key: HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkList\Profiles\?* [targeting certain values]

reg date	reg-UTC	subkey	ProfileName	NameType	DateLastConnected	DateCreated
10/23/2013	02:56:23.515	{2CAEA5..29996ED8}	talonne3	wireless (71)	.. [10/22/2013 22:56:23.515]	.. [08/30/2013 23:45:28.909]
10/21/2013	18:19:12.446	{61659E..C7DBD427}	LOT38	wireless (71)	.. [10/21/2013 14:19:12.446]	.. [10/21/2013 14:19:12.442]
10/14/2013	07:17:57.775	{4A96E5..421A1054}	GCOMM	wireless (71)	.. [10/14/2013 03:17:57.775]	.. [10/14/2013 02:26:35.375]
10/14/2013	06:26:13.551	{A444E3..6104A1C6}	cah-guest	wireless (71)	.. [10/14/2013 02:26:13.551]	.. [10/14/2013 02:26:13.535]
10/14/2013	06:24:12.356	{FB29E5..EC487670}	prg.aero-free	wireless (71)	.. [10/14/2013 02:24:12.356]	.. [10/14/2013 02:24:12.353]
10/13/2013	10:23:59.878	{8D2965..CE346A04}	angelo	wireless (71)	.. [10/13/2013 06:23:59.878]	.. [10/05/2013 11:06:30.894]
10/05/2013	07:13:06.229	{33F09C..B187F34C}	tmobile	wireless (71)	.. [10/05/2013 03:13:06.229]	.. [10/05/2013 03:13:06.229]
...						

Artifact: Network Signatures

Registry key: HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkList\Signatures\?* \?* [targeting certain values]

reg date	reg-UTC	subkey	ProfileGuid	Description	DnsSuffix	DefaultGatewayMac
10/21/2013	18:19:12.444	Unmanaged\010103000.0904f63	{61659..7DBD427}	LOT38	key.chillispot.info	68 7f 74 29 eb 5a
10/14/2013	06:26:35.375	Unmanaged\010103000.6dc31ca	{4A96E..21A1054}	GCOMM	gcomm.cz	d4 ca 6d 80 c4 b3
10/14/2013	06:26:13.551	Unmanaged\010103000.40a8f07	{A444E..104A1C6}	cah-guest	prg.aero	00 10 db ff 20 70
10/14/2013	06:24:12.354	Unmanaged\010103000.2573249	{FB29E..C487670}	prg.aero-free	prg.aero	2c 6b f5 1c 6a 46
10/05/2013	15:06:30.904	Unmanaged\010103000.f8a9318	{8D296..E346A04}	angelo	<none>	00 0c 42 9b 8f 39
10/05/2013	07:13:06.229	Unmanaged\010103000.5f8f95f	{33F09..187F34C}	tmobile	<none>	00 00 0c 07 ac 35
10/03/2013	11:48:58.215	Unmanaged\010103000.d6a2995	{167B2..3F0D3CF}	gogoinflight	<none>	00 e0 4b 22 96 d9
09/29/2013	18:33:06.926	Unmanaged\010103000.16bf8f4	{64534..6D91806}	Samarkand	home	00 26 62 4b 5e b2
09/23/2013	19:18:06.243	Unmanaged\010103000.a9fc2bc	{39E72..044FAD9}	attwifi	landrysmcc.dca.waypo..	00 90 fb 43 a3 88

6.11 Volume Related Artifacts found in Software Hives

- HKLM\Software\Microsoft\Dfrg\Statistics
- HKLM\Software\Microsoft\Windows Search\VolumeInfoCache

Example: *cafae -hive software_hive -volumes > out.txt*

6.12 System Restore Related Artifacts found in Software Hives

- HKLM\Software\Microsoft\Windows NT\CurrentVersion\SystemRestore
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\SPP\Clients

Example: *cafae -hive software_hive -restore > out.txt*

6.13 Web Browsing Related Artifacts found in Software Hives

- HKLM\Software\Microsoft\Internet Explorer
- HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects
- HKLM\Software\Microsoft\Internet Explorer\Toolbar
- HKLM\Software\Wow6432Node\Microsoft\Internet Explorer\Toolbar
- HKLM\Software\Microsoft\Internet Explorer\Explorer Bars
- HKLM\Software\Wow6432Node\Microsoft\Internet Explorer\Explorer Bars
- HKLM\Software\Microsoft\Internet Explorer\Extensions
- HKLM\Software\Wow6432Node\Microsoft\Internet Explorer\Extensions
- HKLM\Software\Microsoft\Internet Explorer\Version Vector
- HKLM\Software\Microsoft\Internet Explorer\Extensions\?*ButtonText
- HKLM\Software\Clients\?*Capabilities\FileAssociations
- HKLM\Software\Clients\?*Capabilities\URLAssociations
- HKLM\Software\Clients\?*shell\open\command
- HKLM\Software\Classes\HTTP\shell\open\command

Example: *cafae -hive software_hive -web > out.txt*

6.14 Service Related Artifacts found in Software Hives

- HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\SharedTaskScheduler
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Explorer\SharedTaskScheduler
- HKLM\Software\Microsoft\SchedulingAgent
- HKLM\Software\Wow6432Node\Microsoft\SchedulingAgent
- HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall\SchedulingAgent
- HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\SchedulingAgent
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks
- HKLM\Software\Microsoft\Windows NT\CurrentVersion\Svchost
- HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Svchost

Example: *cafae -hive software_hive -services > out.txt*

6.15 Persistence Related Artifacts found in Software Hives

- All the keys from *-runkeys* option
- All the keys from the *-shell_spawn* option
- All the keys from the *-inprocservers* option
- A portion of the keys from the *-web* option
- A portion of the keys from the *-explorer* option

Example: *cafae -hive software_hive -persistence > out.txt*

6.16 All Software artifacts

One can parse artifacts from a software hive using the *-all_software* switch.

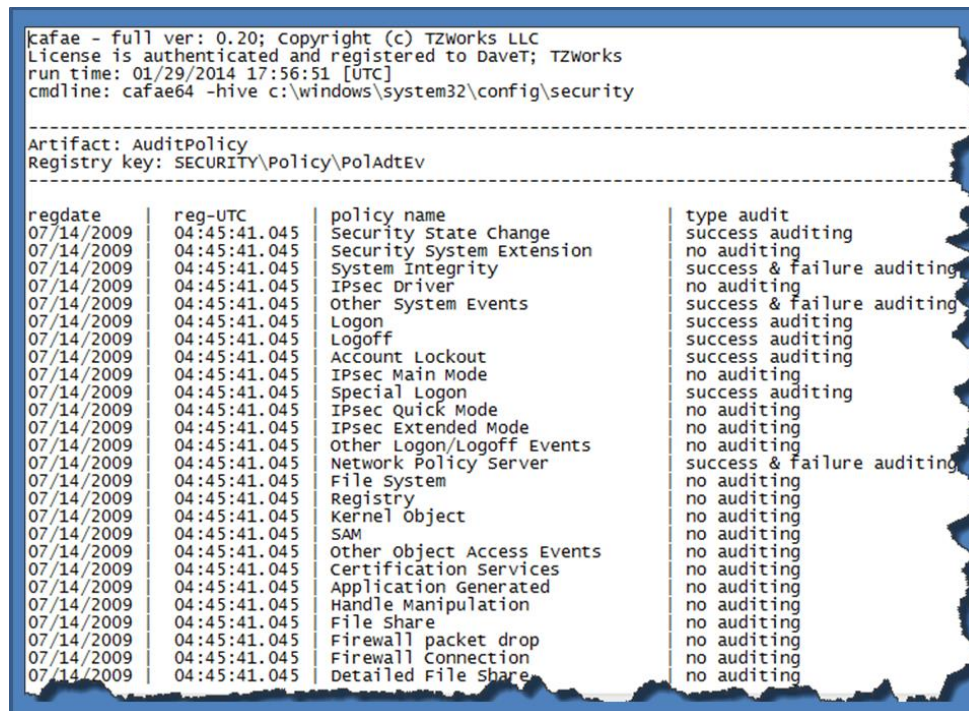
7 Security hive Artifacts

One can parse artifacts from a security hive using the **-all_security** switch.

The security hive artifacts analyzed and reported on includes:

- *HKLM\SECURITY\Policy\PolAcDmS*
- *HKLM\SECURITY\Policy\PolAdtEv*

The report generated contains a breakout of the security policies that are set. Currently, this is only readable when using it with the long (non-csv) output.



regdate	reg-UTC	policy name	type audit
07/14/2009	04:45:41.045	Security State Change	success auditing
07/14/2009	04:45:41.045	Security System Extension	no auditing
07/14/2009	04:45:41.045	System Integrity	success & failure auditing
07/14/2009	04:45:41.045	IPsec Driver	no auditing
07/14/2009	04:45:41.045	Other System Events	success & failure auditing
07/14/2009	04:45:41.045	Logon	success auditing
07/14/2009	04:45:41.045	Logoff	success auditing
07/14/2009	04:45:41.045	Account Lockout	success auditing
07/14/2009	04:45:41.045	IPsec Main Mode	no auditing
07/14/2009	04:45:41.045	Special Logon	success auditing
07/14/2009	04:45:41.045	IPsec Quick Mode	no auditing
07/14/2009	04:45:41.045	IPsec Extended Mode	no auditing
07/14/2009	04:45:41.045	Other Logon/Logoff Events	no auditing
07/14/2009	04:45:41.045	Network Policy Server	success & failure auditing
07/14/2009	04:45:41.045	File System	no auditing
07/14/2009	04:45:41.045	Registry	no auditing
07/14/2009	04:45:41.045	Kernel Object	no auditing
07/14/2009	04:45:41.045	SAM	no auditing
07/14/2009	04:45:41.045	Other object Access Events	no auditing
07/14/2009	04:45:41.045	Certification Services	no auditing
07/14/2009	04:45:41.045	Application Generated	no auditing
07/14/2009	04:45:41.045	Handle Manipulation	no auditing
07/14/2009	04:45:41.045	File Share	no auditing
07/14/2009	04:45:41.045	Firewall packet drop	no auditing
07/14/2009	04:45:41.045	Firewall Connection	no auditing
07/14/2009	04:45:41.045	Detailed File Share	no auditing

8 Sam Hive Artifacts

There are 2 options to process artifacts with the SAM hive. The first just pulls the account information, such as: (a) password reset date, (b) username, (c) encrypted password hashes, (d) lockout time (if present), user ID, group ID, etc. These can be parsed using the **-all_sam** switch.

The Sam hive artifacts analyzed and reported on includes:

- *HKLM\SAM\SAM\Domains\Account\Users*
- *HKLM\SAM\SAM\Domains\Builtin\Aliases\?*IC*

- *HKLM\SAM\SAM\Domains\BuiltIn\Aliases\Members*

Below is a truncated report to show the type of user account and logon stats that are given.

reg date [UTC]	pw reset date [UTC]	rid	username	full name	nt hash	lockout [UTC]	user id	group id
2013/06/02 03:57:45	07/26/2012 07:27:03	0x0001f4	Administrator		616e3dd0	06/02/2013 03:57:44	0x0001f4	0x000201
2013/06/02 03:18:34		0x0001f5	Guest				0x0001f5	0x000201
2013/10/22 16:38:08	08/10/2013 03:03:23	0x0003e9	Donald		96a08e	10/22/2013 16:38:07	0x0003e9	0x000201
2013/10/22 09:05:47	10/01/2013 18:51:20	0x0003eb	HomeGroupUser\$		f38d033	10/22/2013 09:05:46	0x0003eb	0x000201

The second option pulls the similar information that the **-all_sam** switch provides, but allows one to decrypt the encrypted hashes. In this case, the System hive is required. This option is the **-pull_hashes** option, which also uses the parameters: **-sam <location of SAM hive>**, and **-system <location of System hive>**. Note: this option does not try to find the plain text version of the password; it just decrypts the encrypted hash. From this decrypted hash, one can use another tool to find the plain text of the password, if desired. Below is the type of output that is produced (verbose mode is shown).

```
"cmdline: cafae64 -pull_hashes -sam c:\temp\win8.dblake.SAM -system c:\temp\win8.dblake.SYSTEM"

rid: 000001f4
regkey mod: 06/02/2013 03:57:45.377 [UTC]
username: Administrator
comment: Built-in account for administering the computer/domain
group: 0220 : Administrators
last pw reset: 07/26/2012 07:27:03.766 [UTC]
acct expires: never
acct: [disabled; normal acct; pw does not expire]
encrypted lanman hash: <null>
encrypted nt hash: 616e3dd0d904d35494f9f4d92597d568c
clear lanman hash: aad3b435b51404eeaad3b435b51404eeaad3b435b51404eeaad3b435b51404ee
clear nt hash: 7123ba936f3699d934b7e3a1577e6a97

rid: 000003e9
regkey mod: 10/22/2013 16:38:07.682 [UTC]
username: Donald
fullname: Donald Blake
internet acct: dblake@asgard-venture-capital.com
name (2nd source): Donald Blake
```

9 AmCache Hive Artifacts

One can parse artifacts from an AmCache hive using the **-all_amache** switch.

The AmCache hive artifacts analyzed and reported on includes:

- *Amcache.hve\Root\File*
- *Amcache.hve\Root\Programs*
- *Amcache.hve\Root\InventoryApplicationFile*
- *Amcache.hve\Root\InventoryDeviceContainer*
- *Amcache.hve\Root\InventoryDevicePnp*
- *Amcache.hve\Root\InventoryDriverBinary*
- *Amcache.hve\Root\InventoryDriverPackage*

- *Amcache.hve\Root\InventoryApplicationShortcut*
- *Amcache.hve\Root\InventoryApplicationFramework*

Some of the above artifacts depend on what version of the Windows operating system is analyzed.

10 Carving Keys from hives [experimental]

10.1 Carving keys

Included with *cafae* are two experimental carving options: (a) *-carve* and (b) *-carve_deleted*. Both options will work with partial or corrupted hives. The *-carve* option will show both good and deleted entries, while the *-carve_deleted* will only display those entries that are deleted. Using these switches will extract registry keys, their last modified timestamp and the offset where they were found.

A good example of a partial hive, are the registry logs that are used by the operating system to record transactions before committing them to the hives. Below is an example of running this option and the type of output it produces.

cmdline: cafae64 -hive c:\Windows\System32\config\SAM.LOG1 -carve_deleted				
reg date	reg-UTC	offset	del	key name
7/14/2009	04:45:46.661	0x00000420	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}
9/11/2014	23:25:51.771	0x000004a8	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM
7/14/2009	04:45:46.661	0x000006e8	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\RXACT
7/14/2009	04:45:46.676	0x00000810	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains
9/11/2014	20:29:12.794	0x00000898	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin
7/14/2009	04:45:46.661	0x00000b18	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Users
7/14/2009	04:45:46.661	0x00000b98	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Users\Names
7/14/2009	04:45:46.661	0x00000c20	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Groups
7/14/2009	04:45:46.661	0x00000ca8	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Groups\Names
9/11/2014	23:25:51.771	0x00000d28	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Aliases
9/11/2014	23:28:18.428	0x00000db0	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Aliases\Names
9/11/2014	23:25:48.152	0x00000e38	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Aliases\Membe
9/11/2014	23:28:18.428	0x00000ec0	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-371AC8D717C7}\SAM\Domains\Builtin\Aliases\Names\

Note, that since this type of file this is really a transaction log, where only pieces of the hive blocks are present.

10.2 Carving Values

An additional option one can use when carving keys (via the *-carve* or *-carve_deleted*), is the *-vals* switch. This tells *cafae* to also try to extract any values it finds associated with the keys that are carved. Running it on the same log as above yields the same keys, but will value data populated.

cmdline: cafae64 -hive c:\Windows\System32\config\SAM.LOG1 -carve_deleted -vals						
reg date	reg-UTC	offset	del	key name	value data	
7/14/2009	04:45:46.661	0x00000420	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}		
9/11/2014	23:25:51.771	0x000004a8	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}		
7/14/2009	04:45:46.661	0x000006e8	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [01 00 00 00 00 00	
7/14/2009	04:45:46.676	0x00000810	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
9/11/2014	20:29:12.794	0x00000898	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	F [02 00 01 00 00 00 00 eb	
7/14/2009	04:45:46.661	0x00000b18	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
7/14/2009	04:45:46.661	0x00000b98	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
7/14/2009	04:45:46.661	0x00000c20	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
7/14/2009	04:45:46.661	0x00000ca8	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
9/11/2014	23:25:51.771	0x00000d28	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
9/11/2014	23:28:18.428	0x00000db0	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
9/11/2014	23:25:48.152	0x00000e38	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	
9/11/2014	23:28:18.428	0x00000ec0	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}		
9/11/2014	23:28:18.428	0x00000f50	*	CMI-CreateHive{C4E7BA2B-68E8-499C-B1A1-3}	(unnamed) [0x00000000]	

One can use the above switches in combination with the **vsenum** tool to enumerate the registry transaction logs from the various volume shadow copies and pipe them into **cafae's** carving option. The first figure shows how to use **vsenum** to enumerate just the log files from volume shadow copy 1. The second figure shows one how to chain this command to the **cafae** command using the **-pipe** and **-carve** options.

```
cmdline: vsenum64 -dir %vss%1\windows\system32\config -filter "*.log*"

\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\BCD-Template.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\COMPONENTS.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\COMPONENTS.LOG1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\COMPONENTS.LOG2
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\DEFAULT.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\DEFAULT.LOG1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\DEFAULT.LOG2
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SAM.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SAM.LOG1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SAM.LOG2
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SECURITY.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SECURITY.LOG1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SECURITY.LOG2
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SOFTWARE.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SOFTWARE.LOG1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SOFTWARE.LOG2
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SYSTEM.LOG
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SYSTEM.LOG1
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SYSTEM.LOG2
```

cmdline: vssenum64 -dir %vss%1\windows\system32\config -filter "*.log*" cafae64 -pipe -carve_deleted -vals								

Artifact: Carve								
Registry key: [carved keys and values]								
Hive Location: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\BCD-Template.LOG								

reg date	reg-UTC	offset	del	key name	value data			
9/12/2014	00:25:23.232	0x00000420	*	System				
9/12/2014	00:25:23.201	0x00000520	*	System\Description	KeyName [BCD00000000]			
9/12/2014	00:25:23.279	0x00000590	*	System\Objects				
9/12/2014	00:25:23.279	0x00000600	*	System\Objects\{9dea862c-5cdd-4e70-acc1-f32b344d4795}				
9/12/2014	00:25:23.263	0x00000678	*	System\Objects\{9dea862c-5cdd-4e70-acc1-f32b344d4795}\Description	Type [0x10100002]			
9/12/2014	00:25:23.279	0x00000710	*	System\Objects\{9dea862c-5cdd-4e70-acc1-f32b344d4795}\Elements				
9/12/2014	00:25:23.279	0x00000780	*	System\Objects\{9dea862c-5cdd-4e70-acc1-f32b344d4795}\Elements\12000002	Element [\EFI\Microsoft\Boot\			
9/12/2014	00:25:23.263	0x00000848	*	System\Objects\{9dea862c-5cdd-4e70-acc1-f32b344d4795}\Elements\12000004	Element [Windows Boot Mana			

Artifact: Carve								
Registry key: [carved keys and values]								
Hive Location: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\windows\system32\config\SYSTEM.LOG1								

reg date	reg-UTC	offset	del	key name	value data			
5/13/2015	17:36:27.626	0x00001b60	*	CMI-CreateHive{2A7FB991-7BBE-4F9D-B91E-7CB51D4737F5}\ControlSet001				
3/22/2015	02:56:55.193	0x00001ca0	*	CMI-CreateHive{2A7FB991-7BBE-4F9D-B91E-7CB51D4737F5}\ControlSet001\servi				
7/14/2009	04:49:01.599	0x00001d08	*	CMI-CreateHive{2A7FB991-7BBE-4F9D-B91E-7CB51D4737F5}\ControlSet001\servi	ProviderGuid [{7b563579-53c8-			
7/14/2009	04:49:05.935	0x00001e90	*	CMI-CreateHive{2A7FB991-7BBE-4F9D-B91E-7CB51D4737F5}\ControlSet001\servi	EventMessageFile [%SystemRo			
6/29/2015	02:53:11.748	0x00002030	*	CMI-CreateHive{2A7FB991-7BBE-4F9D-B91E-7CB51D4737F5}\ControlSet001\Conti				

If one cannot use the **-pipe** option, one can use the experimental **-enumdir** option, which has similar functionality with more control. The **-enumdir** option takes as its parameter the folder to start with. It also allows one to specify the number of subdirectories to evaluate using the **-num_subdirs <#>** sub-option.

10.3 Hive Statistics

One can pull statistics from a hive or transaction log backing a hive via the **-stats** switch. This option dumps some of the header information from the hive and carves out all the keys (normal keys and any deleted keys that are found). The purpose of the key carving is to generate a histogram of the frequency of keys modified over time. This allows a quick view into which periods of time the registry was most and least active. The histogram time unit shows activity in 1 day increments. Below is a sample output of analyzing the software hive transaction log.


```
"cmdline: cafae64 -hive c:\windows\System32\config\software.log1 -stats"
Hive Transaction Log: Version 1.5
Name: SOFTWARE
Location: c:\windows\System32\config\software.log1
Last Mod Date: 07/28/2015 00:44:39.103 [UTC]
Seq# s: 0000ee39 : 0000ee39
Last known size: 0x07ddf000 [131985408] bytes
First key: 0x00001020 [4128] offset
Checksum: 0x7aed4411
Valid Keys: 517
Deleted Keys: 45
```

Date [UTC]-sort	valid Keys	Deleted Keys
2016/03/08	1	
2015/07/28	25	3
2015/07/27	9	2
2015/07/22	3	
2015/07/20	6	1
2015/07/17		1
2015/07/15	16	
2015/06/10	2	
2015/05/14	17	
2015/05/13	44	
2015/04/15	2	2
2015/04/04	7	
2015/03/21	1	
2015/02/26	2	
2015/02/14	1	
2015/02/12	1	
2015/02/04	1	
2015/01/14		8
2014/12/10		1
2014/11/29	1	
2014/10/15	5	
2014/10/09	2	
2014/10/07	3	
2014/10/06	1	
2014/09/12	96	5
2014/09/11	97	1
2011/04/12	2	
2010/11/21	31	
2009/07/14	141	21

11 Merging Transactional Log files into its Associated Hive (Experimental)

Transaction logs are used to enhance the reliability of the Windows operating system when updating the registry files. Basically, these log files are journals that record the registry data that is to be updated prior to the OS actually committing the final writes to the registry hive. These logs can have the following extensions: .LOG, .LOG1 and .LOG2. The LOG1 and LOG2 extensions are when Windows is in a dual logging mode (and applies to the newer versions of Windows) and the .LOG is when Windows is in the single logging mode. Many times, when looking in the directory containing the hive, one will see all three extensions present.

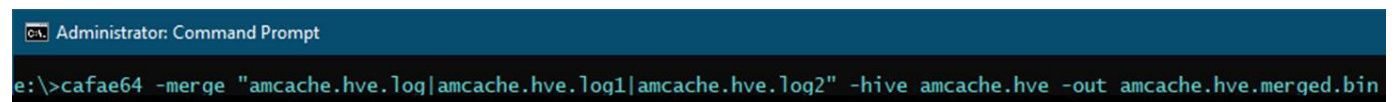
The terminology “dirty” can be used to designate when the hive is out of date and needs to be updated. Conversely, the term “clean” can be used to designate that the hive has been updated and is in sync with all the data. To determine when a registry hive is ‘dirty’ and needs to be updated from the transaction log(s) information, one can look at either the associated hive’s checksum or the primary and secondary sequence numbers or both. An incorrect checksum usually implies the hive is corrupted and a mismatch of the primary and secondary sequence numbers implies the registry hive needs to be updated with the transaction log data.

When analyzing a random transaction log, one can usually see there are multiple sequence numbers associated with interspersed records. This is because a transaction log is constantly updated with new information and when it runs into a space boundary uses a wrap technique to overwrite older records. The sequence number is the indicator used to help order the information so it can be updated properly

to the appropriate hive accurately. Therefore, when updating a registry hive, the algorithm keys off the secondary sequence number of the hive to determine which records from the log are appropriate to accurately update the hive. Therefore, when using the merge option with **cafae**, the tool may determine that no updates are required for a number of reasons: (a) the hive is clean and not dirty, or (b) the hive is dirty, but the appropriate sequence number for updates were not found in one or more of the log files.

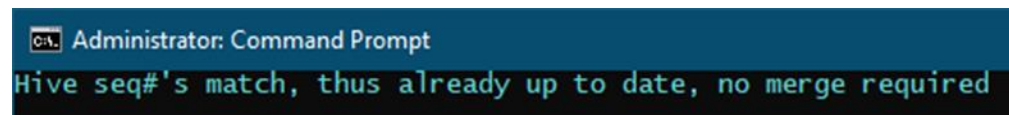
To see the statistics of a hive, one can use the **-stats** option in **cafae**. This will do a number of things, but of importance to this section, are the primary and secondary sequence numbers. If these sequence numbers match, then there is no need to perform the merge option with **cafae**.

To use the merge option, it is advisable to include all log files in the command line. In this way, **cafae** will be able to parse each of the log files, order all the data extracted by sequence number and only act on those records that are appropriate for the update. Below is an example of doing this:



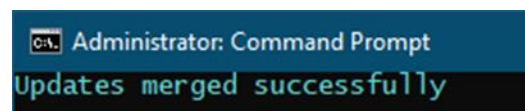
```
Administrator: Command Prompt
e:\>cafae64 -merge "amcache.hve.log|amcache.hve.log1|amcache.hve.log2" -hive amcache.hve -out amcache.hve.merged.bin
```

If the target hive is already up to date, the following message is displayed.



```
Administrator: Command Prompt
Hive seq#'s match, thus already up to date, no merge required
```

If the target hive is dirty and can be updated, then the following message will be displayed if the merge is successful.



```
Administrator: Command Prompt
Updates merged successfully
```

12 Comparing Hives – Experimental

One of the newer options is the ability to difference 2 hives. This option is useful when trying to determine the changes over time between an older hive snapshot with that of a newer one.

The output will only show the changes, whether they are deleted, added or modified entries. The algorithm only looks at changes of subkeys, value names and value data. It does **not** look at changes in: (a) subkey timestamp changes, (b) entries in slack space or (c) registry metadata, such deltas in offsets, etc.

Each change is shown on a separate CSV type delimited line, identifying if there was a new subkey, value added or modified. One can either use the **-csv** or **-csv12t** options for output. This is particularly useful when trying to determine the changes made to a hive prior to an installation of some software and after the installation.

To compare a hive from two different timeframes, the syntax is **-diff "hive1|hive2"**. Below is an example of running this compare option against 2 ntuser.dat hives. The output shows the 'change type' field as either: (a) *new key*, (b) *new value*, (c) *modified value*, (d) *deleted key* or (e) *deleted value*.

The first timestamp shown is the original last modified timestamp. The second timestamp in the next field only appears if there is a modified value, and in which case, it is the timestamp associated with the modification that took place. The 'data' and 'prev data' fields are the values for the latest data, and if it was a modification, the previous data as well.

cmdline: cafae64 -diff user\user1.dat user\user2.dat -out test.csv -dateformat yyyy-mm-dd						
reg date [UTC]	changed [UTC]	change type	subkey	value name	data	prev data
2020-05-03 14:33:10.409		new_key	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Edge\BrowserExitCodes			
2020-05-03 14:33:10.409		new_value	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Edge\BrowserExitCodes	11124-132329851	0x00000000	
2020-05-03 13:12:49.379	2020-05-03 14:33:10.343	modified_value	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Edge\StabilityMetrics	user_experience	0x00000001	0x00000000
2020-05-03 01:14:28.546	2020-05-03 14:33:07.406	modified_value	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Notepad	iWindowPosX	0x00000045	0xfffffa80
2020-05-03 01:14:28.546	2020-05-03 14:33:07.406	modified_value	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Notepad	iWindowPosY	0x00000040	0x0000005a
2020-05-03 01:14:28.546	2020-05-03 14:33:07.406	modified_value	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Notepad	iWindowPosDX	0x0000037c	0x00000305
2020-05-03 01:14:28.546	2020-05-03 14:33:07.406	modified_value	HKEY_CURRENT_USER\ntuser.dat\Software\Microsoft\Notepad	iWindowPosDY	0x00000239	0x000001ec
2019-08-14 04:00:31.321		deleted_key	HKEY_CURRENT_USER\ntuser.dat\System\CurrentControlSet\Control\DeviceCo			
2019-08-14 04:00:31.321		deleted_key	HKEY_CURRENT_USER\ntuser.dat\System\CurrentControlSet\Control\DeviceCo			
2019-08-14 04:00:31.321		deleted_key	HKEY_CURRENT_USER\ntuser.dat\System\CurrentControlSet\Control\DeviceCo			
2019-08-14 04:00:33.100		deleted_key	HKEY_CURRENT_USER\ntuser.dat\System\CurrentControlSet\Control\DeviceCo			
2019-08-14 04:00:33.100		deleted_value	HKEY_CURRENT_USER\ntuser.dat\System\CurrentControlSet\Control\DeviceCo (unnamed)			

There are a couple of sub-options with the **-diff** option, and they are: **-retain_order** which says to the tool to keep the ordering of the files compared in the same order passed in. Otherwise, without this option, the tool internally tries to determine the oldest hive and put that hive first and then compares it to the newest hive. The algorithm to determine oldest to newest is down by examining the hive's sequence numbers. The other sub-option is **-diff_metadata**, in which the output is annotated with the name of the files compared along with their respective sequence numbers and MD5 hashes.

13 Scripting cafae – Experimental

As new registry artifacts are discovered, or as certain artifacts become more important to some clients over others, it would be nice to use **cafae** to just parse those new artifacts without any coding experience.

cafae has always had an internal quasi-scripting interface that can translate specially formatted strings to tell the engine which registry *subkeys* to analyze and/or which values to extract. Up until now, it just has not been available as an option for the outside user.

The two main options include: (a) quick parse using the **-key** option, and (b) parsing using a user defined template via the **-cmdfile** option. The first is useful when exploring the registry, while the second is useful for repeatable analysis by running a script that has been vetted by a qualified security team.

These options are labeled as experimental for two reasons. The first is that they are geared more for the tinkerer or reverser, and it was not meant to be used for the occasional user. The second reason is these options will most likely to be in the beta mode for a long time. This is not because they are new, since the internal engine has been around since the tool was created; it is because the engine is always evolving as new requirements are generated. The main downfall of experimental options is there is less time in the development cycle spent on checking whether various combinations of conflicting options causes boundary conditions.

The benefit, on the other hand, is that once you create a script that extracts the data you want, in the format you want, then it is very easy to give that to a less experienced person to pull the desired data in a repeatable manner.

13.1 Command line Quick Parse of any Key/Value – Experimental

Before getting into User Defined Templates, the easiest way to enumerate a set of child keys, or child values, is to use the **-key** option. This is useful when investigating a certain key, or set of values, that peaks your interest and there is not a canned template to use for the parsing. To use this option, one needs to specify the hive one wishes to operate on via the **-hive <registry hive>**. Then, one needs to specify which subkey to look at via the **-key <subkey to analyze>**. If one wants to enumerate the children keys, then the command would look like:

-key <subkey to analyze> -hive <registry hive> [-enumkeys]

Below is an example of enumerating the children subkeys for the Software hive at the root of the hive. The default behavior will sort by date and show the most recently changed entry first.

```

C:\Windows\system32\cmd.exe

C:\>cafae64 -hive c:\dump\reg.test\win7\SOFTWARE -key HKLM\Software

-----
Registry key: Software [enumkeys]
Hive Location: c:\dump\reg.test\win7\SOFTWARE
-----

reg date      | reg-UTC      | regpath
07/28/2014    | 13:35:12.797 | Software\Microsoft
07/28/2014    | 13:34:43.685 | Software\Wow6432Node
07/24/2014    | 19:11:14.879 | Software\Classes
07/11/2014    | 11:34:16.708 | Software\RegisteredApplications
07/03/2014    | 12:35:46.054 | Software\VMware, Inc.
05/17/2014    | 03:19:52.243 | Software\Clients
03/18/2014    | 02:19:21.046 | Software\NVIDIA Corporation
02/17/2014    | 12:22:07.242 | Software\Debug

```

To enumerate values, one leaves off the option **-enumkeys**, since value enumeration is the default. Below is an example of enumerating the values in the *HKLM\System\CurrentControlSet\Services\dhcp* subkey and redirecting the output to a text file. The timestamp is from the parent subkey which is the same for all the values, and hence, the time is the same. The 'value name' and 'value data' columns

reflect the child values. Note: when the data is binary, only the beginning chunk of hex bytes is displayed.

```
cmdline: cafae64 -hive c:\dump\reg.test\win7\SYSTEM -key HKLM\System\CurrentControlSet\Services\dhcp
-----
Registry key: System\CurrentControlSet\Services\dhcp [enumvalues]
Hive Location: c:\dump\reg.test\win7\SYSTEM
-----
```

reg date	reg-UTC	value name	value data
07/14/2009	04:53:19.368	ServiceDll	%SystemRoot%\system32\dhcpcore.dll
07/14/2009	04:53:19.368	DisplayName	@%SystemRoot%\system32\dhcpcore.dll,-100
07/14/2009	04:53:19.368	Group	TDI
07/14/2009	04:53:19.368	ImagePath	%SystemRoot%\system32\svchost.exe -k LocalServiceNetworkRestricted
07/14/2009	04:53:19.368	Description	@%SystemRoot%\system32\dhcpcore.dll,-101
07/14/2009	04:53:19.368	ObjectName	NT Authority\LocalService
07/14/2009	04:53:19.368	ErrorControl	1
07/14/2009	04:53:19.368	Start	2
07/14/2009	04:53:19.368	Type	32
07/14/2009	04:53:19.368	DependOnService	NSI; Tdx; Afd
07/14/2009	04:53:19.368	ServicesSidType	1
07/14/2009	04:53:19.368	RequiredPrivileges	SeChangeNotifyPrivilege; SeCreateGlobalPrivilege
07/14/2009	04:53:19.368	FailureActions	80 51 01 00 00 00 00 00 00 00 00 00 03 00 00 00 ... [28 more bytes]

For more complex queries, one can use the wildcard option within the subkey path to tell **cafae** to pull subkeys that match the desired pattern. When using wildcards, the number of records returned can be massive. For this reason, **cafae** offers a filter to pull only those value names you are interested in. For example, to pull all the services, but only extract the following values: “start and imagepath” one could issue the following command line in screen dump below. Note: the entire command needs to be on one line. It was edited for the output below to span two lines to fit the graphic on the screen. Secondly, the names passed into the extract option are treated as case insensitive.

By specifying a wildcard (via the ‘?’ in the path) or the extract option, (via **-extract “start | imagepath”**), a couple of things happen: (a) the script engine allows for multiple subkey names per value data and thus annotates the output with the subkey that equates to the position of the wildcard in the path, and (b) the sort is also adjusted to look at the subkey name and use that as opposed to the date. The extract option also adjusts the output so each extraction name specified is now its own separate column.

```
cmdline: cafae64 -hive c:\dump\reg.test\win7\SYSTEM -key HKLM\System\CurrentControlSet\Services\?*
-extract "Start | ImagePath"
-----
Registry key: System\CurrentControlSet\Services\?* [targeting certain values]
Hive Location: c:\dump\reg.test\win7\SYSTEM
-----
```

reg date	reg-UTC	subkey	Start	ImagePath
07/04/2014	12:23:21.999	1394ohci	0x0003	system32\DRIVERS\1394ohci.sys
07/04/2014	12:23:21.671	ACPI	0x0000	system32\drivers\ACPI.sys
07/04/2014	12:23:21.640	AFD	0x0001	\SystemRoot\system32\drivers\afd.sys
07/03/2014	12:35:16.779	ALG	0x0003	%SystemRoot%\System32\alg.exe
07/03/2014	12:35:16.779	AcpiPmi	0x0003	\SystemRoot\system32\drivers\acpipmi.sys
07/03/2014	12:35:16.779	AeLookupSvc	0x0003	%systemroot%\system32\svchost.exe -k netsv
07/03/2014	12:35:16.779	AmdK8	0x0003	\SystemRoot\system32\drivers\amdK8.sys
07/03/2014	12:35:16.779	AmdPPM	0x0003	\SystemRoot\system32\drivers\amdppm.sys
07/03/2014	12:35:16.779	AppID	0x0003	\SystemRoot\system32\drivers\appid.sys
07/03/2014	12:35:16.779	AppIDSvc	0x0003	%SystemRoot%\system32\svchost.exe -k Loca
07/03/2014	12:35:16.779	AppMgmt	0x0003	%SystemRoot%\system32\svchost.exe -k nets
07/03/2014	12:35:16.779	Appinfo	0x0003	%SystemRoot%\system32\svchost.exe -k nets
07/04/2014	12:25:48.083	AsyncMac	0x0003	system32\DRIVERS\asyncmac.sys
07/03/2014	12:35:16.779	AudioEndpointBuilder	0x0002	%SystemRoot%\System32\audio\endpointbuilder.exe -k Loc

Aside from the normal name extraction, there are two reserved names for value extraction that the script engine will interpret as special, and perform the actions in the table below.

Reserved Name	Meaning
(unnamed)	Pulls the value data for 'unnamed' fields
(other fields)	Pulls all the other value data and groups it into one column

If you want to sort, filter, or translate the data further, one can append qualifiers to each of the extraction names. Some of the basic qualifiers are as follows:

Qualifier	Type	Meaning
!KP	Sort	Key to sort on – sort the data on this value
!R	Filter	Required field – meaning value must be present to display this record
!WP	Filter	Wildcard – meaning name is a partial string and match would be a partial match
!TF	Translate	Assume data is FILETIME and translate it to time UTC
!TU	Translate	Assume data is Unix time and translate it to time UTC
!TO	Translate	Assume date is OLE time and translate it to time UTC
!BG	Translate	Convert bytes to GUID string
!BU	Translate	Convert bytes to Unicode string
!BV	Translate	Convert bytes to value

To take the previous example of extracting values, if one wanted to sort, on the “start” key, which has values 0, 1, 2, 3, 4, etc, one would use the following extract phrase in the script: via **-extract “start!K / imagepath”**. The **!K** instructs the script engine that the value name of “start” is the key to sort on.

In some cases you want to only pull out entries that have a certain value that is populated. You can tell the script engine to do that via the qualifier **!R**. Again, taking the previous example, one could say they only want service keys that contained a start value that is populated. This would be done via the command: via **-extract “start!K!R / imagepath”**. The **!R** and **!K** qualifiers are mutually exclusive, so it is up to the user whether or not to keep them together. With this example, the script engine would only pull entries that had a “start” value populated, and then sort the entries on the “start” value. Below is an example:

```
cmdline: cafae64 -hive c:\dump\reg.test\win7\SYSTEM -key HKLM\System\CurrentControlSet\Services\?*
-extract "start!K!R | imagepath"

-----
Registry key: System\CurrentControlSet\Services\?* [targeting certain values]
Hive Location: c:\dump\reg.test\win7\SYSTEM
-----
```

reg date	reg-UTC	subkey	start	imagepath
07/04/2014	12:23:21.671	ACPI	0x0000	system32\drivers\ACPI.sys
07/04/2014	12:23:21.640	CLFS	0x0000	System32\CLFS.sys
07/04/2014	12:23:21.640	CNG	0x0000	System32\Drivers\cng.sys
07/04/2014	12:23:22.280	Disk	0x0000	system32\drivers\disk.sys
07/04/2014	12:23:21.640	Fileinfo	0x0000	system32\drivers\fileinfo.sys
07/04/2014	12:23:21.640	FltMgr	0x0000	system32\drivers\fltMgr.sys
07/04/2014	12:23:21.640	storflt	0x0000	system32\drivers\vmstorfl.sys
07/04/2014	12:23:21.640	vdrvroot	0x0000	system32\drivers\vdrvroot.sys
07/04/2014	12:23:21.640	vmci	0x0000	system32\DRIVERS\vmci.sys
07/04/2014	12:23:21.640	volmgr	0x0000	system32\drivers\volmgr.sys
07/04/2014	12:23:21.640	AFD	0x0001	SystemRoot\system32\drivers\afd.sys
07/04/2014	12:23:21.640	Beep	0x0001	
07/04/2014	12:23:21.640	CSC	0x0001	system32\drivers\csc.sys
07/04/2014	12:23:21.640	Dfsc	0x0001	System32\Drivers\dfsc.sys
07/04/2014	12:23:21.640	Mstfs	0x0001	
07/04/2014	12:23:21.640	NetBIOS	0x0001	system32\DRIVERS\netbios.sys

The next option is the **-level <number of levels to traverse>** option, which specifies the number of levels you want **cafae** to look before ending. Keep in mind, the more levels you look at the longer it takes and the more memory that is required. Below is an example that traverses many shell objects and pulls out the handler used. Notice, in addition to **-level 1** option, the wildcard in the path option was used, as well as the **-extract "(unnamed)"** option. The resulting output gives one all the **CLSIDs** of the **ContextMenuHandlers**.

```
cmdline: cafae64 -hive c:\dump\reg.test\win7\SOFTWARE -key HKLM\Software\Classes\?*ShellEx\ContextMenuHandlers\?*
-level 1 -enumvalues -extract (unnamed)

-----
Registry key: Software\Classes\?*ShellEx\ContextMenuHandlers\?* [targeting certain values]
Hive Location: c:\dump\reg.test\win7\SOFTWARE
-----
```

reg date	reg-UTC	subkey	(unnamed)
07/14/2009	04:53:38.073	*\shellx\contextmenuhandlers\briefcasemenu	{858BD920-42A0-1069-A2E4-08002B30309D}
07/24/2014	19:11:14.925	*\shellx\contextmenuhandlers\epb	{09A47860-11B0-4DA5-AFA5-26D86198A780}
07/14/2009	04:53:38.073	*\shellx\contextmenuhandlers\open with	{09799AFB-AD67-11d1-ABCD-00C04FC30936}
07/14/2009	04:53:38.073	*\shellx\contextmenuhandlers\open with encryptionmenu	{A470F8CF-A1E8-4f65-8335-227475AA5C46}
07/14/2009	04:53:38.073	*\shellx\contextmenuhandlers\sharing	{F81E9010-6ea4-11ce-a7ff-00aa003ca9f6}
07/11/2014	11:34:18.323	*\shellx\contextmenuhandlers\snagitmainshellx	{CF74B903-3389-469c-B3B6-0204D204FCBD}
12/04/2013	15:30:34.947	*\shellx\contextmenuhandlers\winzip	{E0D79304-84BE-11CE-9641-444553540000}
07/14/2009	04:53:38.073	*\shellx\contextmenuhandlers\{90aa3a4e-1cba-4233-b8bb-535773d48449}	Taskband Pin
07/14/2009	04:53:38.073	*\shellx\contextmenuhandlers\{a2a9545d-a0c2-42b4-9708-a0b2badd77c8}	Start Menu Pin
07/14/2009	04:53:38.088	AllFilesystemObjects\shellx\contextmenuhandlers\copyaspathmenu	{f3d06e7c-1e45-4a26-847e-f9fcdee59be0}
07/14/2009	04:53:38.276	DesktopBackground\shellx\contextmenuhandlers\desktopslideshow	{7BA4C740-9E81-11CF-99D3-00AA004AE837}
07/14/2009	04:53:38.276	Directory\shellx\contextmenuhandlers\encryptionmenu	{0bf754aa-c967-445c-ab3d-d8fda9baefef}
07/24/2014	19:11:14.925	Directory\shellx\contextmenuhandlers\epb	{A470F8CF-A1E8-4f65-8335-227475AA5C46}
04/12/2011	08:28:45.245	Directory\shellx\contextmenuhandlers\offline files	{09A47860-11B0-4DA5-AFA5-26D86198A780}
07/14/2009	04:53:38.276	Directory\shellx\contextmenuhandlers\sharing	{474C98EE-CF3D-41f5-80E3-4AAB0A804301}
07/11/2014	11:34:18.324	Directory\shellx\contextmenuhandlers\snagitmainshellx	{F81E9010-6ea4-11ce-a7ff-00aa003ca9f6}

13.2 User Defined Templates (or cmdfiles) - Experimental

To get more sophisticated, one can wrap the command line script options into a file, which is called a *"User Defined Template"*. This is simply a text file generated in notepad identifying any combination of registry paths/keys you are interested in. Using the syntax rules, one can add comments and have various instructions to tell **cafae** to parse a hive in different ways. The template can be used in combination with other arguments passed on the command line to specify additional output formatting instructions and which hive to operate on.

This is what we use internally to perform regression testing and to test out new artifacts before committing them to the canned reports in **cafae**. While extremely useful for us, since we know its limitations and boundary conditions, this option may prove to be premature for public release, since it is

still rough around the edges for the new user, and hence, the name *experimental* at the top of this section.

Below are the guidelines and rules to use this option.

13.2.1 Template Rules

As mentioned above, the templates are just text files, so they can be generated with any text editor. Care must be taken to ensure that extra control characters are not inserted into the template files. Having extra control characters will negatively affect the template parsing engine. For this reason, it is recommended that a simple text editor be used when editing a template file.

The parsing rules for these templates are as follows:

1. General Rules
 - a. Each line is parsed separately.
 - b. A line that starts with a double forward slash (eg. //) is ignored and used for comments
 - c. A blank line is ignored
 - d. Any line not satisfying rule (1b) and (1c) above is assumed to be a command
 - e. All command lines are in CSV format, where the separator is a comma. This applies to commands with parameters. So if a command has a keyword and argument(s), then the keyword is listed, then a comma, then an argument, then another comma, then the next argument. This simple rule allows all the keywords and arguments to be separated.
2. Command Lines

Must start with the sequence: **!cmd**, and the entire command must be on one line.

- a. The command sequence can contain the following options, using comma delimiters (in any order):
 - enumreg**
 - key, < registry key path of parent key to operate on>**
 - level, < number of levels down to evaluate from the parent key>**
 - enumvalues [or -enumkeys]**
 - sort_by_name [or -sort_by_date]**
 - name <name of this artifact>**
- b. There are many other switches that can be used, but are not published here due to the experimental nature of them. If you would like to use templates, contact us and we can help you generate one.

13.2.2 Mapping Template parameters to Log2Timeline Output

If using the **-csv/2t** option, one can map the custom commands in a template file to Log2Timeline's output. The following guidelines are used.

- **-name** <name of artifact> equates to "**sourcetype**" field
- **-append_subkey_auto** equates to "**short**" field. The internal script engine will look at the location of any wildcards and use that offset as the subkey to use as its data. If no wildcards exist, then it will see if any levels to traverse are greater than 0. If there are, then it will use the offset of last segment of the subkey to use as its data.
- **-enumvalues** or **-enumkeys** equates to name[data] pairs in "**desc**" field. More discussion on how to control which values are extracted are discussed in the next section.
- **-hive** <name of hive> equates to "**filename**" field.
- **-key** <registry key path> equates to "**extra**" field.
- **-user** <username> equates to "**user**" field. [note, the **-user** command is not within the template but part of the command line options for **cafae** and can be used in conjunction with the **-cmdfile** option]
- **-hostname** <name> equates to "**host**" field. [note, the **-hostname** command is not within the template, but part of the command line options for **cafae** and can be used in conjunction with the **-cmdfile** option]

13.2.3 Template Examples

There are some differences between the command line script examples and the templates that exist in files. As stated earlier, the line needs to be preceded by the characters: **!cmd**, **-enumreg**,. Also, everything is comma delimited; the commands as well as the command arguments. If a command can take multiple arguments, then the arguments need to be delimited with a pipe character. Finally, you should not to hardcode the **-hive <path to registry hive>** as part of the template command. That portion of the information will come from the command line directly and will be more important when you pipe in multiple hives for **cafae** to process.

Below are some examples of various commands that can be used in a template:

13.2.3.1 Example 1: Pull Services

```
// Pull services that have a start key
!cmd, -enumreg, -key, HKLM\SYSTEM\CurrentControlSet\Services\, -enumvalues, -extract,
Start!R!K | Type | ImagePath | ServiceDll, -level, 2, -comment, Services with start
key, -append_subkey_auto
```

13.2.3.2 Example 2: Pull Mounted Devices (using a custom option designed for Mounted Devices)

```
// using the generic options
!cmd, -enumreg, -key, HKLM\SYSTEM\MountedDevices\, -enumvalues, -comment, Mounted
Devices
```

```
// using a custom option designed for the Mounted Devices data.
!cmd, -enumreg, -key, HKLM\SYSTEM\MountedDevices\, -enumvalues, -comment, Mounted
Devices, -mounted_devices
```

13.2.3.3 Example 3: Pull many of the Shell spawning entries

```
!cmd, -enumreg, -key,
HKLM\SOFTWARE\Classes\?*Shell\?*Command, -enumvalues, -extract,
(unnamed)!R, -comment, Shell spawning entries
```

Once the command is generated and saved as a text file, one can invoke the template file via the command:

cafae -hive <location of registry hive> -cmdfile <path of file>

14 Converting Segmented CSV formats into Database Friendly Formats

When running **cafae** to pull many artifacts from a hive into one results file, the CSV output will vary depending on artifact that is processed. While the *-bodyfile* and *-csv/2t* formats will preserve the CSV structure, the default CSV output will show the results as segmented CSV sections. Each CSV section will represent a different artifact type. This can create problems when trying to import the **cafae** results into other databases for analysis.

To solve this problem, one can use the **csvdx** tool to take the segmented CSV results (or any CSV results) and convert the artifact output it into either JSON or SQLite. See the **csvdx** webpage (https://tzworks.com/prototype_page.php?proto_id=34) and/or user guide (<https://tzworks.com/prototypes/csvdx/csvdx.users.guide.pdf>).

15 Handling Corrupt Hives

There are cases when one will come across corrupt hives or be able to partially reconstruct a hive from another tool. One case where this happens frequently is the reconstruction of hives from a memory capture (reference the Volatility plugin *dumpregistry*). In some cases, the desired hive(s) can be reconstructed. In other cases, however, the desired hive(s) may have portions of the data paged out (eg. not in memory, but only on disk) at the time of the memory dump. In these cases, the reconstruction of the hive(s) would be incomplete and can cause registry parsers to crash. While we continue to strive to make **cafae** handle corrupt hives, there are always boundary conditions that come up. Having said that, if any corrupt hives cause **cafae** to crash, please send them to us so we can improve the robustness of **cafae** in this area.

16 List of Options

Option	Extra	Description
-hive		Use this option to specify which hive to process artifacts from. Syntax is -hive <hive file>
-livehives		List the local user hives available on the target machine.
-pull_hashes	*	Given a SAM and SYSTEM hive via the options -sam <hive> and -system <hive> will extract the encrypted hashes and decrypt them. Doesn't try to compute the text password, but only shows the unencrypted hashes of the password. Syntax is -pull_hashes -sam <hive> -system <hive>
-showkeys	*	Display the registry keys extracted using the specified option. Syntax is -showkeys [artifact option]
-showcmds	*	Display the internal script used for the specified option. Syntax is -showcmds [artifact option] . This can assist users in developing their own custom scripts by seeing how we have used the internal scripting engine.
-all_software		Extract all software unique artifacts. Applies to the <i>software</i> hive.
-all_system		Extract all system unique artifacts. Applies to the <i>system</i> hive.
-all_security		Extract all security unique artifacts. Applies to the <i>security</i> hive.
-all_amcache		Extract all AmCache unique artifacts. Applies only to Win 8 and later
-all_sam		Extract all SAM unique artifacts.
-stats	*	Experimental option. Quick view of registry stats including histogram of activity
-scan_size	*	Scans registry entries locating those that are at or above a specified threshold size. The syntax is -scan_size <minimum size> .
-scan_entropy	*	Scans registry entries locating those that have an entropy value that is specified or higher. Values above 80 through 95 yield useful results to give an indication which entries have randomized data in them (which can either imply compression or encryption, among other formats). The syntax is -scan_entropy <percent entropy> .

-carve	*	Experimental option. Given any hive or partial hive, this option will try to extract keys. Useful option if the hive is corrupted. If desiring to extract values as well, use the -vals switch in conjunction with the option.
-carve_deleted	*	Experimental option. Given any hive or partial hive, this option will try to extract all deleted keys. If desiring to extract values as well, use the -vals switch in conjunction with the option.
-merge	*	Experimental. Takes one or more log files and merges them into associated base hive. The syntax is -merge "<log1> <log2>" -hive <orig hive> -out <new_hive> . Algorithm checks to ensure log files have the proper sequence number entries available prior to merging.
-diff	*	Experimental. Takes 2 or more hives and diffs them outputting the delta changes. The syntax is -diff "hive1 hive2" . Internally, this option will try to compare the oldest hive to the latest hive. If that is not desired, one can force the order of comparison by using the sub-option -retain_order . Sub-options allowed for formatting the output include: -csv and -csv12t . If wishing to see which files are being compared with their sequence numbers, use the sub-option: -diff_metadata .
-key	*	Experimental option that allows the user to parse a specified registry subkey path from a specified hive quickly. Since this option has a number of nested options, and consequently allows one to specify many options at once, cafae may become unstable, depending if various options are passed into it that are in conflict. Also, since this is still beta, the format most likely will change in future releases. The intent is to allow one to enumerate the child values or keys. The default is to enumerate values. To enumerate keys, use the option [-enumkeys]. The -level switch allows one to go zero or more levels deep. The default is 0 levels (which means, just the first level). The [-extract "val1 val2 ... valN"] option allows one to specify which value names to extract to the output. The syntax is -key <subkey path> -hive <reg hive> [-enumkeys] [-level <#>] [-extract "arg1 arg2 ... argX"]
-cmdfile	*	Option that allows the user to customize which registry artifacts to extract as well as which fields to output. The syntax is -cmdfile <filename> .
-csv		Outputs the data fields delimited by commas. Since filenames can have commas, to ensure the fields are uniquely separated, any commas in the filenames get converted to spaces.

-csvl2t		Outputs the data fields in accordance with the log2timeline format.
-bodyfile		Outputs the data fields in accordance with the 'body-file' version3 specified in the SleuthKit. The date/timestamp outputted to the body-file is in terms of UTC. So if using the body-file in conjunction with the mactime.pl utility, one needs to set the environment variable TZ=UTC.
-base10		Ensure all size/address output is displayed in base-10 format vice hexadecimal format. Default is hexadecimal format.
-username		Option is used to populate the output records with a specified username. The syntax is -username <name to use> .
-hostname		Option is used to populate the output records with a specified hostname. The syntax is -hostname <name to use> .
-userstats		Pulls username account information from hive [ntuser.dat only] and populates output with the extracted username.
-pipe		Used to pipe files into the tool via STDIN (standard input). Each file passed in is parsed in sequence.
-enumdir	*	Experimental. Used to process files within a folder and/or subfolders. Each file is parsed in sequence. The syntax is -enumdir <folder> -num_subdirs <#> .
-filter	*	Filters data passed in via STDIN via the -pipe or -enumdir options. The syntax is -filter <"*.ext *partialname* ..."> . The wildcard character '*' is restricted to either before the name or after the name.
-no_whitespace		Used in conjunction with -csv option to remove any whitespace between the field value and the CSV separator.
-notrunc	*	The default behavior is to truncate large datasets and only show the first part of the data (applies to binary data). If you want change the default behavior, then use this option to not use any truncation. Doing this, however, may not render your data in a usable format.
-csv_separator		Used in conjunction with the -csv option to change the CSV separator from the default comma to something else. Syntax is -csv_separator "/" to change the CSV separator to the pipe character. To use the tab as a separator, one can use the -csv_separator "tab" OR -csv_separator "\t" options.
-dateformat		Output the date using the specified format. Default behavior is -dateformat

		" <i>yyyy-mm-dd</i> ". Using this option allows one to adjust the format to mm/dd/yy, dd/mm/yy, etc. The restriction with this option is the forward slash (/) or dash (-) symbol needs to separate month, day and year and the month is in digit (1-12) form versus abbreviated name form.
<i>-timeformat</i>		Output the time using the specified format. Default behavior is <i>-timeformat "hh:mm:ss.xxx"</i> One can adjust the format to microseconds, via <i>"hh:mm:ss.xxxxxx"</i> or nanoseconds, via <i>"hh:mm:ss.xxxxxxxxxx"</i> , or no fractional seconds, via <i>"hh:mm:ss"</i> . The restrictions with this option is a colon (:) symbol needs to separate hours, minutes and seconds, a period (.) symbol needs to separate the seconds and fractional seconds, and the repeating symbol 'x' is used to represent number of fractional seconds. (Note: the fractional seconds applies only to those time formats that have the appropriate precision available. The Windows internal filetime has, for example, 100 nsec unit precision available. The DOS time format and the UNIX 'time_t' format, however, have no fractional seconds). Some of the times represented by this tool may use a time format without fractional seconds, and therefore, will not show a greater precision beyond seconds when using this option.
<i>-pair_datetime</i>	*	Output the date/time as 1 field vice 2 for csv option
<i>-quiet</i>		Don't show the progress in the output
<i>-all_controlsets</i>	*	If processing a system hive, this option will tell <i>cafae</i> to look at all the <i>ControlSets</i> versus just the default <i>ControlSet</i> (called the <i>CurrentControlSet</i>).

17 Authentication and the License File

This tool has authentication built into the binary. The primary authentication mechanism is the digital X509 code signing certificate embedded into the binary (Windows and macOS).

The other mechanism is the runtime authentication, which applies to all the versions of the tools (Windows, Linux and macOS). The runtime authentication ensures that the tool has a valid license. The license needs to be in the same directory of the tool for it to authenticate. Furthermore, any modification to the license, either to its name or contents, will invalidate the license.

17.1 *Limited* versus *Demo* versus *Full* in the tool's Output Banner

The tools from *TZWorks* will output header information about the tool's version and whether it is running in *limited*, *demo* or *full* mode. This is directly related to what version of a license the tool authenticates with. The *limited* and *demo* keywords indicates some functionality of the tool is not available, and the *full* keyword indicates all the functionality is available. The lacking functionality in the *limited* or *demo* versions may mean one or all of the following: (a) certain options may not be available, (b) certain data may not be outputted in the parsed results, and (c) the license has a finite lifetime before expiring.

18 References

1. yaru - Yet Another Registry Utility, www.tzworks.com
2. sbag - ShellBag Parser, www.tzworks.com
3. MiTec Registry Analyzer, by Allan S Hay, 12/2004
4. SANs Institute. Forensics 408 course (Jan 2010)
5. Windows Registry Forensics, Advanced Digital Forensic Analysis of the Windows Registry, by Harlan Carvey, Syngress, 2011
6. Various MSDN articles
7. SleuthKit [Body-file](http://wiki.sleuthkit.org) format, <http://wiki.sleuthkit.org>
8. Log2timeline CSV format, <http://log2timeline.net/>
9. UserAssist focus time and count, reference: <http://zoltandfw.blogspot.com/2012/10/userassist.html>