March 2020

Windows ShellBags Forensics

Investigative value of Windows ShellBags

LIFARS

your digital world, **secured**

Contents

Overview 3
Purpose and forensic value of Windows ShellBags 3
Location of ShellBag data 3
ShellBag creation, structure and interpretation 5
BagMRU key 5
MRUListEx value Explained 6
Windows registry time-related settings and their interpretation.....6
Timestamp "anomalies" originating in filesystem 8
Parsing ShellBags with ShellBags Explorer.....8
Evidence inferred from ShellBags 9
ShellBags and external drives: FAT vs NTFS 11
What if… 12
Parsing ShellBags with RegRipper 17
Conclusion 17
References 18

OVERVIEW

Windows ShellBags are one of the well-known and valuable sources of information regarding computer system's user behavior. Although their primary purpose is to improve user experience and "remember" preferences while browsing folders, information stored in ShellBags can be critical during forensic investigation.

PURPOSE AND FORENSIC VALUE OF WINDOWS SHELLBAGS

Windows ShellBags were introduced into Microsoft's operating system Windows 7, and are still present on all Windows 10 system releases. Generally, speaking ShellBags are designed to hold information about user's preferences while browsing folders. That means that if the user changes folder view from "Large Icons" to, for example, "Details", the settings get stored in ShellBag.

When you open, close or change viewing option of any folder on your computer, either from Windows Explorer, or from the Desktop (even by right-clicking or renaming the folder), a ShellBag record is created or updated. This implies the following:

1. If any directory is mentioned in Windows ShellBags, it must have been present on the system at some time – even if it is not present anymore. This is valid for local filesystem including compressed archives, as well as for network locations (e.g. remote mapped shares) and removable devices (e.g. USB flash drive).

2. As these actions and viewing preferences are tied to the user's registry hives, we can connect specific user account and specific folder. Moreover, we can get information about when the folder has been last accessed, from MAC timestamps contained in ShellBags.

LOCATION OF SHELLBAG DATA

Most forensics investigators are familiar with incredible amount of information one can get from user's registry hives, namely, **NTUSER.DAT**. Not everyone, however, is as well acquainted with **USRCLASS.DAT**, additional hive, storing valuable data about users. While investigating ShellBags, both **NTUSER.DAT** and **USRCLASS.DAT** registry hives are needed.

ShellBag information is divided into 2 groups:

- Information about folders recently accessed via Explorer that can be found in:
    - USRCLASS.DAT\Local Settings\Software\Microsoft\Windows\Shell\Bags,
    - USRCLASS.DAT\LocalSettings\Software\Microsoft\Windows\Shell\BagMRU,
    - USRCLASS.DAT\LocalSettings\Software\Microsoft\Windows\ShellNoRoam\Bags and,
    - USRCLASS.DAT\LocalSettings\Software\Microsoft\Windows\ShellNoRoam\BagMRU

- Information about folders accessed from Desktop that can be found in:
    - NTUSER.DAT\ Software\Microsoft\Windows\Shell\Bags and NTUSER.DAT\ Software\Microsoft\Windows\Shell\BagMRU,
    - NTUSER.DAT\ Software\Microsoft\Windows\ ShellNoRoam\Bags and,
    - NTUSER.DAT\ Software\Microsoft\Windows\ ShellNoRoam\BagMRU

Therefore, to begin a ShellBags investigation, we first must locate these registry hives for all users present on the system (or, if doing targeted investigation, hives specific to suspect user/s only). NTUSER.DAT is located under:

C:\Users\<username>\ a.k.a. %userprofile%, and USRCLASS.DAT can be found in C:\Users\<username>\AppData\Local\Microsoft\Windows

At the time of writing this article there is little information about any forensically valuable data that can be gathered from the **...\ShellNoRoam\...** hives. Therefore, we won't mention them in this article.

SHELLBAG CREATION, STRUCTURE AND INTERPRETATION

While browsing filesystem via Explorer, ShellBag entries are created the first time the folder is opened. Each directory is given a number, starting with 0. ShellBag entries are modified when the user changes folder view preferences. This means that the last write time of the registry key may be time of last manipulation with corresponding folder[1]. After opening the child directory, a ShellBag entry is added to the right side in the corresponding branch and is assigned a number - incrementing so far the highest assigned number among child directories of the same parent by one.

BAGMRU KEY

When looking at ShellBags, the root directory is represented by the topmost BagMRU key. Its subkeys represent child-bags and subdirectories placed in the parent directory. They are referred by numeric value – 0, 1, 2, …, while their binary entries store details such as path to the folder and long and short name of directory. Thus, we can reconstruct directory structure as if we were looking at it in Explorer.

In the figure 1, below we can see how ShellBags look like in the registry. The structure of ShellBags corresponding to the C: drive is expanded. Entry for C:\ is the topmost "0" subkey right below BagMRU subkey. Each of the "0" subkeys represent one subdirectory of the C:\ drive. For example, child "0" stands for "Users" directory, next "0" denotes user's home folder, and its subkeys are this user's directories. Entry "2" contains long and short name of "!Install" directory. To take a closer look at additional data, stored in ShellBag entries, additional parsing is necessary.

[1] For deeper information on when are ShellBag entries created and modified, please check Resources in the end of this document. Excessive explanation is beyond scope of this work.

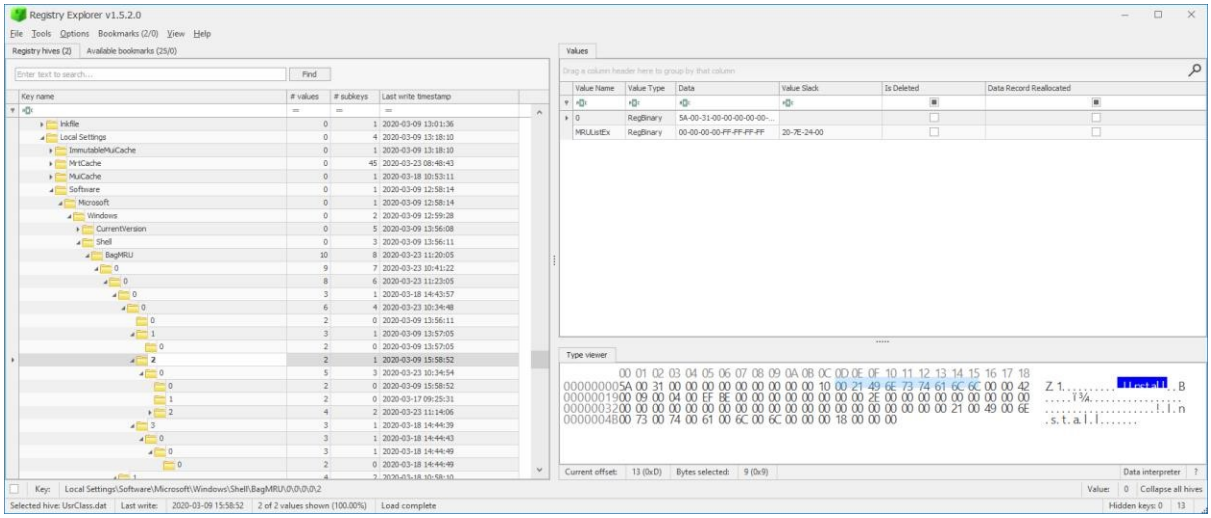

Figure 1. ShellBags in Windows registry

MRULISTEX VALUE EXPLAINED

Each key in BagMRU has a value MRUListEx. It tells us about which subdirectories were last accessed. Let's inspect the example below:

Registry file: C:\Users\User\AppData\Local\Microsoft\Windows\UsrClass.dat
Key: Local Settings\Software\Microsoft\Windows\Shell\BagMRU\0
Last write: 2020-03-24 17:45:50
Value: MRUListEx (RegBinary)
Data: 00-00-00-00-02-00-00-00-07-00-00-00-04-00-00-00-06-00-00-00-03-00-00-00-01-00-00-00-05-00-00-00-FF-FF-FF-FF

The data is a string, in which each 4 bytes represent a number corresponding to Bag entry, i.e., to the last opened subdirectory of this folder. In this case, the last opened subdirectory has a corresponding Bag 0, second last opened was 2, third was Bag 7 etc. When a ShellBag is opened, MRUListEx values shift to the right and most recently opened value is written to the leftmost position. All timestamps in the tree will be rewritten to the same timestamp.

We will use this behavior of MRUListEx for revealing additional information later.

WINDOWS REGISTRY TIME-RELATED SETTINGS AND THEIR INTERPRETATION

When beginning a new investigation, time zone settings of the investigated system need to be discovered and recorded. This is important because various artifacts store timestamps in differently. For example, Windows registry key's LastWrite time is

interpreted in UTC[2]. As we will mention later, some of timestamp data written in ShellBags come from $MFT. In $MFT, again, all MACB timestamps[3] is stored in same manner as in registry, so no time shifts are necessary. When timestamp is about to be written into ShellBags, it is first converted to UTC time.

Knowledge of the time zone is necessary to correlate registry information with artifacts with timestamps stored in local time. For example, we could take the last modify time of suspicious Prefetch file as it is displayed in Windows Explorer as estimated last run time of the corresponding executable. Let this be 24th March 2020, 13:45:10; the time zone is UTC-5. If we want to check for a suspicious ShellBag entry, carrying information about new folder browsed at that time, we need to consider that ShellBag registry data is stored in UTC. Therefore, look at period around 18:45, as this is the corresponding UTC time. It is also important to check if our parsing tool presents the investigated artifact in UTC, or in local time set on investigator's system (or if it allows the investigator to adjust output so that it is presented in time zone of originating device).

Time zone settings are stored in SYSTEM registry hive in the following folder: **C:\Windows\System32\config.** Precise location of the time zone subkey is located here: **SYSTEM\ <CurrentControlSet>\Control\TimeZoneInformation**. The value of **CurrentControlSet** can easily be looked up in the **SYSTEM \Select** key. If the value Current is set to one, the respective path will be located here: **SYSTEM\ControlSet001\Control\TimeZoneInformation**. Value TimeZoneKeyName holds a string with standard time zone naming (e.g. Central Europe Standard Time), and Bias/Daylight Bias denotes difference between current time zone and UTC, in minutes. They also tell if daylight saving time is set.

Another time-related data to consider is stored in the registry key here: **SYSTEM\<CurrentControlSet>\Services\W32Time\Parameters**. Value NtpServer reveals name of the server, with which the investigated host synchronizes the clock. Type has 3 possible entries: NTP stands for manually configured NTP server synchronization, Nt5DS is default standing for domain-level sync, and NoSync means that the device does not synchronize its system clock with any external system at all.

If the clock is not synced correctly, any timestamps gathered from the system will be misleading, no matter if they were collected from the registry, $MFT or other. This may not cause that much trouble in case of isolated networks, as all hosts will have same time skew and the overall timeline will not be that affected. However, once a host with the wrong clock connects to the system with a differently synced clock (no matter if

[2] To be precise, registry LastWrite timestamps are stored in FILETIME structure, which represents the number of 100-nanosecond intervals since January 1, 1601. A such, they carry no information about time zone. This 64-bit value is convertible to human readable datetime format.
[3] (File Last) Modified, (File Last) Accessed, (Metadata) Changed (i.e. $MFT Entry Changed), and Birth (i.e. File Created) timestamps

external host has "correct" time, of if it is in same time zone), any artifacts produced by mutual interaction may be severely affected. For example, email send/receive times may be inconsistent, or file timestamps may differ on both systems without directly changing them.

TIMESTAMP "ANOMALIES" ORIGINATING IN FILESYSTEM

Since Windows 2000, the default filesystem on Windows-based PC has been NTFS. On Windows XP, FAT32 has been quite widespread, and this filesystem is still used for e.g. USB flash disks. ExFAT and ReFS are other filesystems that can occur during investigation. However, these filesystems handle timestamps differently and therefore, these should be considered in ShellBags investigations.

FAT timestamps are stored using the local time of the computer where they were generated and they offer lower precision. This is most markable in the Last Access timestamp, as it only shows accuracy of days.

PARSING SHELLBAGS WITH SHELLBAGS EXPLORER

After a necessary intro into time zone settings and a few words on timestamps in different filesystems, we will proceed with ShellBags investigation.

There are multiple freeware tools available for parsing ShellBags; one (or actually 2) are authored by Eric Zimmerman. The first, ShellBags Explorer comes with GUI, which offers comfortable user experience and is handy when processing smaller amounts of data. His command-line cousin, SBECmd.exe, allows you to process multiple user registry hives at a time and produces csv output. Both tools support live registry processing, allowing you to process registry of a machine you are running the tool on. In some scenarios, you can process exported registry hives offline. Note if you are processing live hives, you need to run ShellBags Explorer with administrative privileges. Registry hives are protected – locked for standard user on running system, and thus not accessible. ShellBags Explorer (both GUI and command line version) has several advantages, compared to other tools in the field. It parses additional timestamps, included in ShellBag entries, MFT Sequence and file record numbers etc.

EVIDENCE INFERRED FROM SHELLBAGS

After opening ShellBags Explorer, load either live registry, or offline captured NTUSER.DAT or USRCLASS.DAT. ShellBags Explorer parses the hives and shows the tree structure of filesystem, as it is seen in Windows Explorer. Besides names of directories and subdirectories it includes multiple timestamps which must be treated with respect to facts mentioned in the previous section - Windows registry timestamps and their interpretation:

1. Registry last write time:
    a. Key's LastWrite timestamp is updated when a value is changed in the Registry key. This corresponds to opening the folder or changing folder view options. [4]

2. First Interacted timestamp:
    a. This information can be inferred indirectly and only for some folders. First interaction can be derived for "bottom-most and right-most" entries in expanded tree structure of BagMRU. Upon opening new folder, its ShellBag entry is added on the right side of its corresponding branch. Therefore, Last Modified time of bottom-most child bag corresponds to first exploring its parent folder.

3. Last Interacted timestamp:
    a. Again, it is not possible to obtain this value without additional information, and it cannot be inferred for every directory. We can use MRUListEx value to see which child entry has been modified as last. Then, LastWrite time of parent key is the time of last interaction with subdirectory at beginning of MRUListEx list.

4. $MFT details:
    a. Target accessed, created and modified timestamps, NTFS sequence number etc. This data is the MAC info of folders, found in and copied out from $MFT at the time of entry creation. Thus, they correspond to MAC times at time of creation of ShellBag entry, and they don't reflect any further modifications of these information in $MFT.

---

[4] Registry key's LastWrite time can be manipulated. It is more difficult than standard timestomping of files and folders, yet not impossible. Investigators should be aware of it when looking at ShellBags – or any other registry keys.

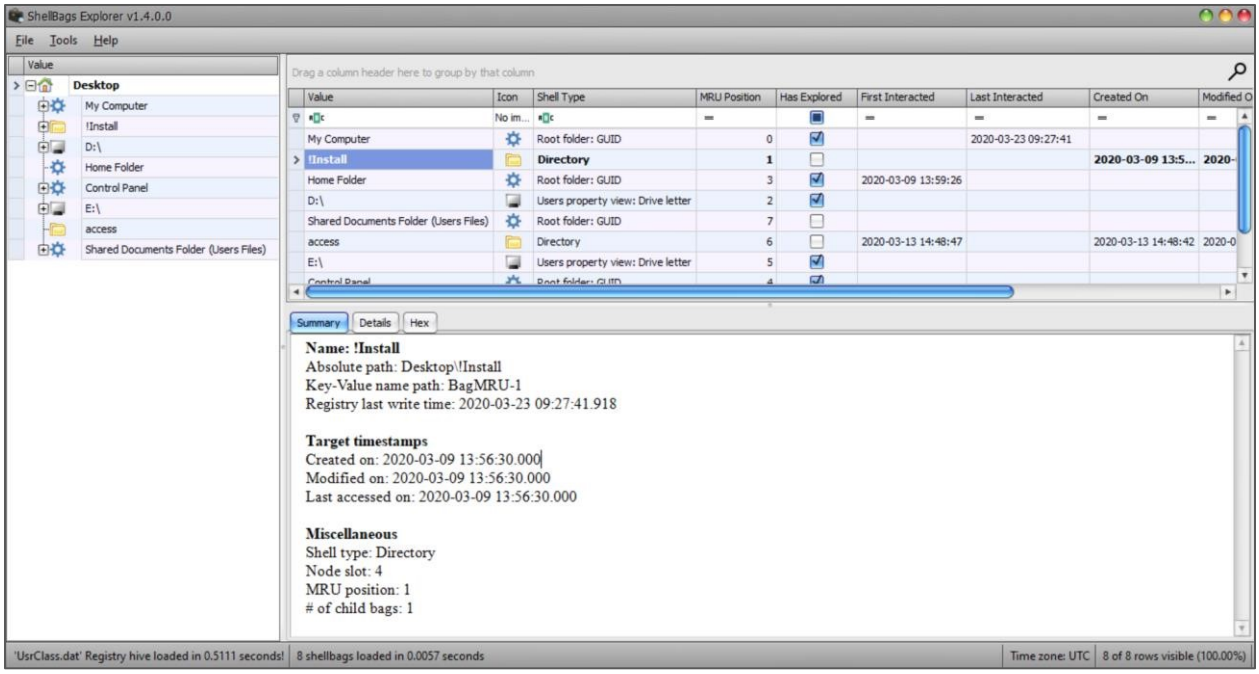

Figure 2. UsrClass.dat in ShellBags Explorer

In figure 2 above, UsrClass.dat registry is parsed using ShellBags Explorer. Note that the First/Last Interacted times are not present for every folder. MRU Position reflects content of MRUListEx entries.

Mapping of ShellBag entries to "Desktop", "Control Panel" and other items or directories in ShellBags Explorer (and other tools as well) is done through mapping values at specific offsets (see hexadecimal view of ShellBag entry above) to type and translation of GUIDs to known items.

For example:

Let the topmost BagMRU "0" have value 0 with hexadecimal content 19-00-2F-43-3A-5C-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00. 2F at 3rd byte indicates this is ShellBag Type of drive letter, following bytes represent path C:\.

If Type at offset 0x02 is 0x1F, we are looking at root folder ShellBag. At offset 0x04 (from Type field) we can find 16-byte GUID. Mapping of well-known GUIDs to names can be searched for online, see References. For example, 20d04fe0-3aea-1069-a2d8-08002b30309d is GUID for "My Computer", in ShellBags seen as 14-00-1F-50-E0-4F-D0-20-EA-3A-69-10-A2-D8-08-00-2B-30-30-9D-00-00.

Figure 3 below presents detailed information about USB drive, mapped as "E:". Note that this thumb drive has not been connected to the system in time of UsrClass.DAT extraction. Despite that, we can learn what filesystem this device used (FAT), what are MAC times of folders it contained, and for some entries, time of first/last interaction. This can be very helpful in cases, for example when a system gets infected through external device insertion.

ShellBags can be very useful when the investigation involves data exfiltration. For example, if a user copies data to the USB thumb drive or to some network location, evidence of accessing such locations may be found in ShellBags, even quite a long time after the incident happened. Information will be preserved in ShellBags, thus providing evidence against the suspect denying knowledge or browsing of these locations. Clearing ShellBags without leaving trace is not easy, and mere absence of ShellBags raises suspicion – and points out higher level of attacker's skills.

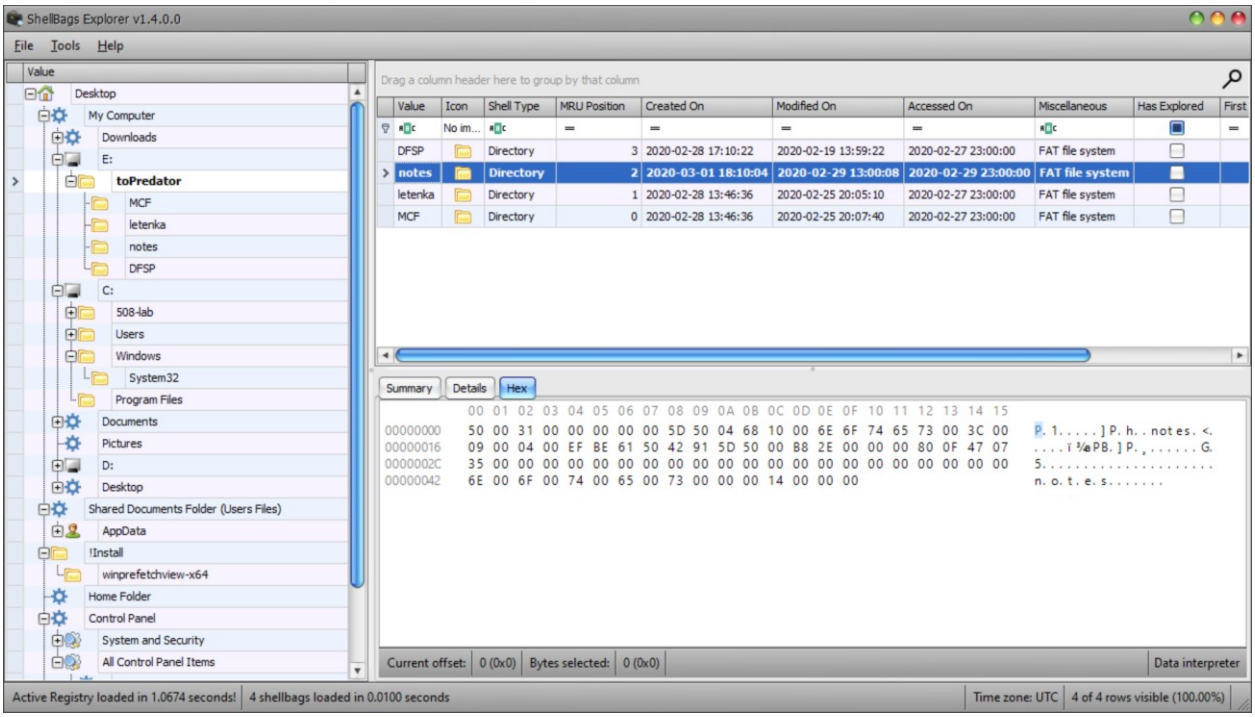

Figure 3. ShellBags from NTUSER.DAT

How does the tool know about the filesystem used an attached drive? Remember already mentioned information contained in BagMRU entries, namely, MFT and inode details. FAT32 filesystem makes sequence numbers equal to 0, compared to NTFS which does assigns every MFT entry a nonzero value of sequence number and file

entry. Similar behavior can be observed with ExFAT devices. Sequence number will be zero. Only file entry will be present for FAT and ExFAT.
To differentiate between FAT and ExFAT, difference in accuracy of timestamps can be leveraged. FAT stores Last Access Time with precision of days, so data like dd.mm.yyyy 12:00:00 UTC will be recorded. ExFAT will provide Last Access Time with precision of 2 seconds.[5]

WHAT IF…

Consider a scenario when a thumb drive is inserted into computer. Folders may be already present there; the user may copy or move some directory from local system to this drive and take them away. Files may be changed on external system, original deleted from USB, replaced by updated folders and copied back to original location. Will these actions influence ShellBag information? What if external system where these files have been modified is in different time zone, and what impact would have NTFS vs. FAT formatted drive? From information shared in previous sections, answers should be deductible. To prove them, we made a few experiments and observed changes from filesystem[6] and ShellBags.

On the "C:" drive of Windows 10 machine (build 18362), we created a folder with several sub- and subfolders. We than copied this folder to FAT32- and NTFS- formatted USB flash drives and took it to external system of the same OS build. We performed several actions, observed their influence on ShellBags and summarized results in tables below.

Local system time zone: UTC +1, local time ~15:00 and later

USB format: FAT32 and NTFS

External system time zone: UTC-5, local time ~10:00 and later

ShellBags Explorer time zone: UTC

[5] In ExFAT, Modify and Create timestamps will have granularity of 10 ms, compared to 2s in FAT. NTFS provides even greater precision of 100ns. However, how precise data we get largely depends on tool used for parsing data.
[6] Times observed on local and external system come from $Standard Information attribute in $MFT.

Copying to FAT32 device

	Directory name	Create (B)	Modify (M)	Access (A)	Note
Local system	C:\test	2020-23-03 12:19	2020-25-03 13:32	2020-25-03 13:32	
USB in local system	D:\test	2020-25-03 15:02	2020-25-03 13:32	2020-25-03 12:00	Directory copied to USB and opened there
USB in external system	E:\test	2020-25-03 10:02	2020-25-03 8:32	2020-25-03 12:00	USB inserted into external system and opened there
External system	D:\test	2020-25-03 10:35	2020-25-03 10:35	2020-25-03 10:35	Files copied to external system
External system	D:\test	2020-25-03 10:35	2020-25-03 10:38	2020-25-03 10:38	Directory modified on external system
Original folder deleted from USB, updated folders copied to USB and transfered to originating system. As the location it is the same, even after copying updated folder creation time did not change.					
USB back to local system	D:\test	2020-25-03 11:10	2020-25-03 10:38	2020-25-03 12:00	Time is from different time zone.
Local system	C:\test	2020-23-03 12:19	2020-25-03 16:16	2020-25-03 12:00	Modify and access time updated from USB

ShellBags Data

	Directory name	TargetCreated	TargetModified	TagretAccessed	Registry LastWrite
SB Local system	C:\test	2020-03-23 11:19:38.000	2020-03-23 11:19:38.000	2020-03-23 11:19:38.000	2020-03-25 12:27:10.249
SB USB in local system	D:\test	2020-03-25 14:02:22.000	2020-03-25 12:32:12.000	2020-03-24 23:00:00.000	2020-03-25 14:09:27.305

SB USB in external system	E:\test	2020-03-25 17:39:06.000	2020-03-25 17:32:12.000	2020-03-25 23:00:00.000	2020-03-25 13:29:25.909 (Viewing preferences change on USB)
SB External system	D:\test	2020-03-25 14:35:08.000	2020-03-25 14:35:08	2020-03-25 14:35:08	2020-03-25 14:35:23.089
Original folder deleted from USB, updated folders copied and transfered to originating system. ShellBags stores original MFT timestamps.					
SB USB back to local system	D:\test	14:02:22.000	2020-03-25 12:32:12.000	2020-03-25 23:00:00.000	2020-03-25 14:09:27.305 (same, because preferences were not changed)
SB back to local system	C:\test	2020-03-23 11:19:38.000	2020-03-23 11:19:38.000	2020-03-23 11:19:38.000	2020-03-25 15:27:31.164

We can observe that the way FAT32 understands time zone produces several strange things while copying folders to and from devices using this filesystem. Time is stored as local, so when we moved from a "earlier" time zone to a "later" one, relying solely on Windows Explorer we can see files and folders created "in the future". However, ShellBags are first converted to UTC and only after that are written into registry. ShellBags timestamps are therefore always in UTC (resp. stored in FILETIME structure).

Despite any change that occurred outside, when folder is copied back to original location (which remained intact in the meantime, so both folders were merged), in ShellBags are still original MAC timestamps associated with that folder, coming from MFT and from time of first browsing this folder. Even when subdirectories were edited outside the system.

Copying to NTFS formatted device

	Directory name	Create(C)	Modify(M)	Access (A)	Note
Local system	C:\test	2020-23-03 12:19	2020-25-03 13:32	2020-25-03 13:32	
USB in local system	E:\test	2020-25-03 15:03 (equals time of copying folder to USB stick)	2020-25-03 15:03	2020-25-03 15:03	Directory copied to USB and opened there
USB in external system	F:\test	2020-25-03 10:03	2020-25-03 10:03	2020-25-03 10:03	USB inserted, folder browsed. Time zone is reflected.
This time we copied folder to new location 2 times. After first copying, we opened the folder so that the ShellBag is created. Than, we deleted it and copied once more.					
External system	C:\test	2020-25-03 9:53	2020-25-03 9:53	2020-25-03 9:53	1st copy to external system and delete folder
External system	C:\test	2020-25-03 10:41	2020-25-03 10:41	2020-25-03 10:41	2nd copy of directory to external system
External system	C:\test	2020-25-03 10:41	2020-25-03 10:42	2020-25-03 10:42	Directory modified on external system
Original folder was deleted from USB, updated folders were copied and transfered to originating system, but this time to new location in the same parent directory C:\.					
USB back to local system	E:\test	2020-25-03 16:15	2020-25-03 16:15	2020-25-03 16:15	NTFS did not preserve local time from other device
Local system	C:\!\test	2020-25-03 16:27	2020-25-03 16:27	2020-25-03 16:27	Time od copying folders

ShellBags Data					
		TargetCreated	TargetModified	TagretAccessed	Registry LastWrite
SB Local system	C:\test	2020-03-23 11:19:38.000	2020-03-23 11:19:38.000	2020-03-23 11:19:38.000	2020-03-25 14:04:21.304 (editing folder preferences)
SB USB in local system	E:\test	2020-03-25 14:03:44.000	2020-03-25 14:03:46.000	2020-03-25 14:03:46.000	2020-03-25 14:11:36.795
SB USB in external system	F:\test	2020-03-25 14:03:44.000	2020-03-25 14:03:46.000	2020-03-25 14:03:46.000	2020-03-25 13:29:25.909
Here we can see that MFT timestamps from first presence of directory are preserved					
SB External system	C:\test	2020-03-25 13:53:20	2020-03-25 13:53:20	2020-03-25 13:53:20	2020-03-25 14:41:46.332
Original folder was deleted from USB, updated folders were copied and transfered to originating system, but this time to new location in the same parent directory C:\.					
SB USB back to local system	E:\test	2020-03-25 14:03:44.000	2020-03-25 14:03:46.000	2020-03-25 14:03:46.000	2020-03-25 15:27:07.271
SB back to local system	C:\!\test	2020-03-25 15:27:22.000	2020-03-25 15:27:22.000	2020-03-25 15:27:22.000	2020-03-25 15:27:24.989

Comparing this with previous experiments with FAT32 USB device, the differences are clear: resolution of Last Access times, and timestamps on USB are not preserving time settings of external system. Again, ShellBags information were converted to UTC and after that written to registry.

One last mystery to be solved: If one time zone is UTC+1 and other is UTC-5 Eastern Time, why is timestamp difference 5 hours, not 6? Answer is in the date: on 25th March, USA already used daylight saving time, although according to clock on system, time zone is still UTC-5. In fact, it is UTC-4. On the originating system, time shift has not taken place yet, and so time zone is true UTC +1.

PARSING SHELLBAGS WITH REGRIPPER

Another well-known tool which parses ShellBags data is RegRipper. The tool, written in Perl is available across different platforms. Plugin shellbags can be run against UsrClass.dat hive to get readable output, like command line version of ShellBagsExplorer.

MRU Time	Modified	Accessed	Created	Zip_Subfolder	MFT File Ref	Resource
(undefined)						My Computer [Desktop\0\]
						My Computer\C:\ [Desktop\0\0\]
						My Computer\C:\Users [Desktop\0\0\0\]
(undefined)						My Computer\C:\Users\User [Desktop\0\0\0\0\]
	2020-03-09 13:56:04	2020-03-09 13:56:04	2020-03-09 12:58:14		4103/3	My Computer\C:\Users\User\Downloads [Desktop\0\0\0\0\0\]
						My Computer\C:\Users\User\Desktop [Desktop\0\0\0\0\2\]
(undefined)						My Computer\C:\Users\User\Desktop\!Install [Desktop\0\0\0\0\2\0\]
	2020-03-09 14:01:58	2020-03-09 14:01:58	2020-03-09 14:01:08		89958/6	My Computer\C:\Users\User\Desktop\!Install\hashmyfiles-x64 [Desktop\0\0\0\0\2\0\0\]
						My Computer\E:\ [Desktop\0\5\]
(undefined)	2020-02-27 19:09:28	2020-02-27 23:00:00	2020-02-28 13:43:16			My Computer\E:\toPredator [Desktop\0\5\0\]
	2020-02-19 13:59:22	2020-02-27 23:00:00	2020-02-28 17:10:22			My Computer\E:\toPredator\DFSP [Desktop\0\5\0\0\]

Table 1. Snippet of RegRipper's ShellBags plugin

CONCLUSION

Windows ShellBags are an invaluable artifact when it comes to user file/folder knowledge investigation. They preserve potentially historical data on what directories have been browsed on the system and allows mapping actions to specific users. Not only it stores data about local folder access, but also network locations, external drives, online sharing etc. We discussed various scenarios to observe how ShellBag information may or may not be affected by usage of different filesystem or changes in time zone. There are several freeware tools available for ShellBags parsing, and commercial forensics suites (EnCase, Magnet AXIOM etc.) parse them as well. ShellBags are

important artifact that every investigator should be well familiar with to make best use of information it can provide.

REFERENCES

Harlan Carvey: Windows Registry Forensics: Advanced Digital Forensic Analysis of the Windows Registry 2nd Edition. Syngress; 2 edition (April 8, 2016)

Vincent Lo: Windows ShellBag Forensics inDepth – Case Study: https://www.sans.org/reading-room/whitepapers/forensics/windows-shellbag-forensics-in-depth-34545. Updated on Nov 19, 2014

Eric Zimmerman's Tools: https://ericzimmerman.github.io/#!index.md

RegRipper github repo: https://github.com/keydet89/RegRipper2.8