ShellBags Explorer

SA Eric R. Zimmerman
Federal Bureau of Investigation
eric.zimmerman@ic.fbi.gov
saericzimmerman@gmail.com
801-514-4064

Last revised: 8/19/2015 10:29:19 AM

Revision history

11/21/2014 Rev. 1 – Initial release

Table of Contents

Requirements 6
What are ShellBags? 6
ShellBags location in the registry 6
Using RegEdit to view ShellBag data 7
ShellBag binary contents 7
MRUListEx 9
Subkeys and their relation to values 9
LastWrite times and their value in ShellBags analysis 11
Why another ShellBags program? 13
Capabilities overview 14
How does it work? 14
Why use ShellBags Explorer and not ShellBag (sbag) Parser by TZWorks, MiTeC Registry Analyser or RegRipper? 14
Getting started 14
ShellBagsExplorer.exe 15
Tree view 16
Left clicking a tree node 16
Right clicking a tree node 16
Grid view 16
Left clicking a grid entry 16
Right clicking a grid entry 16
Details view 16
Hex view 16
Status bar 16
Menus 17
File 17
Tools 17
Help 18
Workflow overview 18
Messages window 20
Tree view 21
Grid view 24

Details view 31
Hex view 34
SBECmd.exe 37
Processing multiple, unrelated hives 38
Processing multiple, related hives 40
Time zone support 40
General usage tips and tricks 42
Version changes 43
Version 0.7.0.0 43
Version 0.6.1.0 43
Version 0.6.0.0 43
Version 0.5.4.0 43
Version 0.5.3.0 43
Version 0.5.2.0 43
Version 0.5.1.0 44
Version 0.5.0.5 44
Version 0.5.0.4 44
Version 0.5.0.3 44
Version 0.5.0.2 45
Version 0.5.0.1 45
Version 0.5.0.0 45
Version 0.4.3.0 45
Version 0.4.2.0 45
Version 0.4.1.0 45
Version 0.3.9.0 46
Version 0.3.8.0 46
Version 0.3.7.0 46
Version 0.3.6.0 46
Version 0.3.5.0 46
Appendix A – Contributors 47
Appendix B – Additional resources 47

Requirements

ShellBags Explorer requires Microsoft .net framework version 4.5.1 full runtime or greater to be installed. It is available at http://www.microsoft.com/en-us/download/details.aspx?id=40779.

What are ShellBags?

ShellBags are of great forensic value from Intrusion investigations to Malware Causality examinations to timeline generation in that they are a perfect example of Locard's exchange principle - "a perpetrator will bring something into a crime scene and leave with something from it."

By design, Microsoft® Windows may track a cyber intruder's actions on a host system after compromise if the actor uses RDP or other Remote Connection controls and/or Windows Explorer to drop binaries onto the system, to access network resources, browse compressed archives, etc. Additionally, all directory traversal done with Windows Explorer is tracked and maintained in the registry. This data includes multiple timestamps and other pieces of information that provide context and can be used to show knowledge and intent when it comes to accessing directories or other resources on a computer.

From the anti-forensics standpoint, an absence of ShellBag entries may suggest system cleaning or overt action by an end-user and potential sophistication of the actor. Lastly, bad actors are becoming smarter every day - not every forensic examination will contain evidence in plain sight/allocated space. ShellBags, along with other artifacts, may point to evidence that existed at one point in time or may assist the examiner in looking at the broader picture when only a piece is known.

ShellBags are a set of Windows Registry keys located in NTUser.dat and USRClass.dat registry hives (primarily usrclass.dat) that maintain view, icon, position, size (and other attributes) of folders when using Windows Explorer. They are likely to persist information even when the original directories, files, and physical devices have been removed from the system and due to this, can serve as a "history" of sorts into data that was previously on a system but may have since been removed. When combined with volume shadow copies (specifically, older copies of registry hives), significant insight into a user's activity can be gleaned.

ShellBags location in the registry

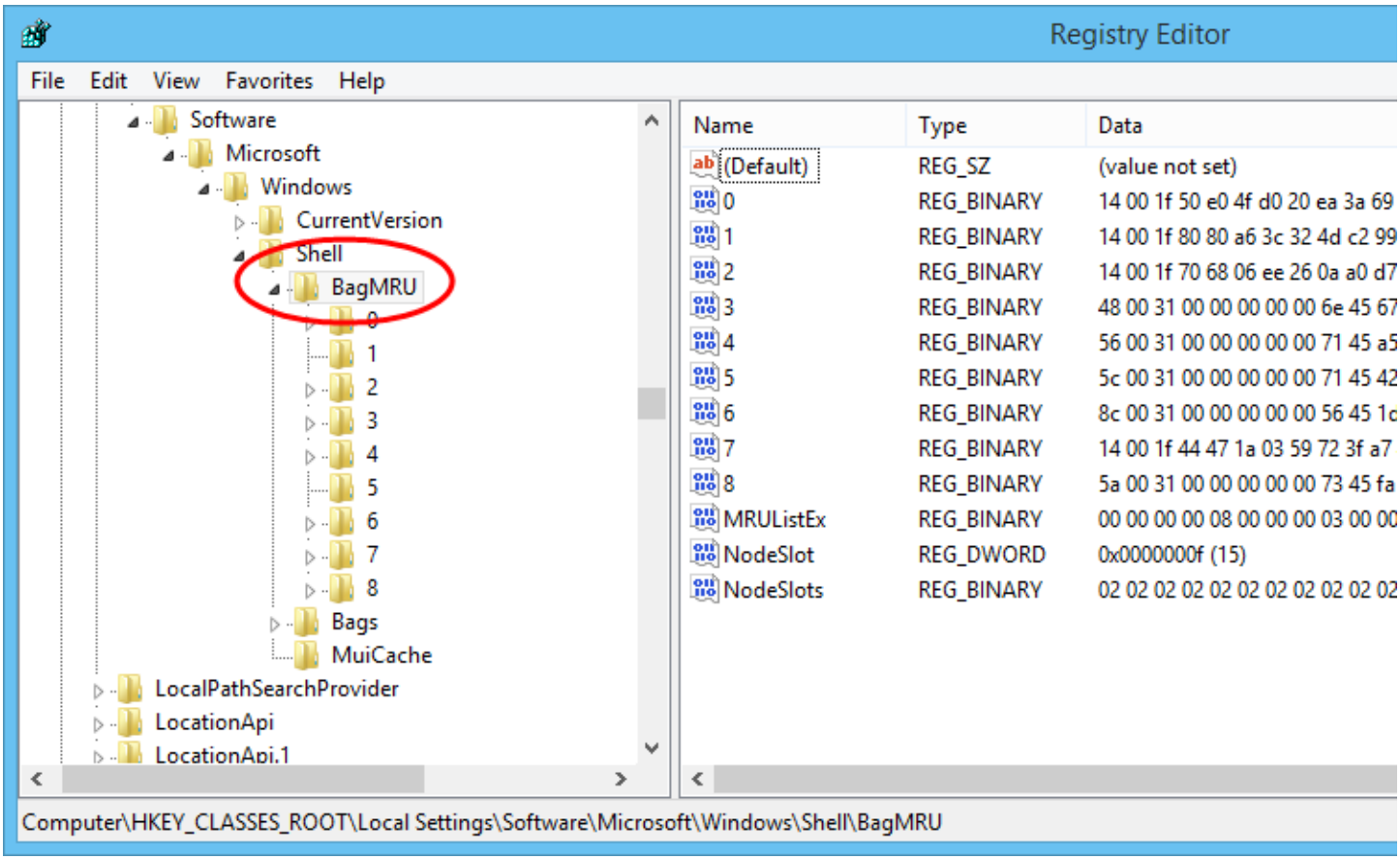
In Windows Vista and newer (including server operating systems based on the same technology), ShellBag data is located in the following registry keys and subkeys:

- HKEY_CURRENT_USER\Software\Microsoft\Windows\Shell\Bags
- HKEY_CURRENT_USER\Software\Microsoft\Windows\Shell\BagMRU (ntuser.dat)
- HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\Bags
- **HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\BagMRU** (ntuser.dat)
- **HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\BagMRU** (userclass.dat)
- HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\Bags

SBE currently looks at the bold keys. The “Bags” keys are not looked at as they have been determined to relate to window position, size, etc,

Using RegEdit to view ShellBag data

The regedit utility included with Windows can be used to view ShellBag data. After starting regedit, navigate to the HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\BagMRU key (shown below).

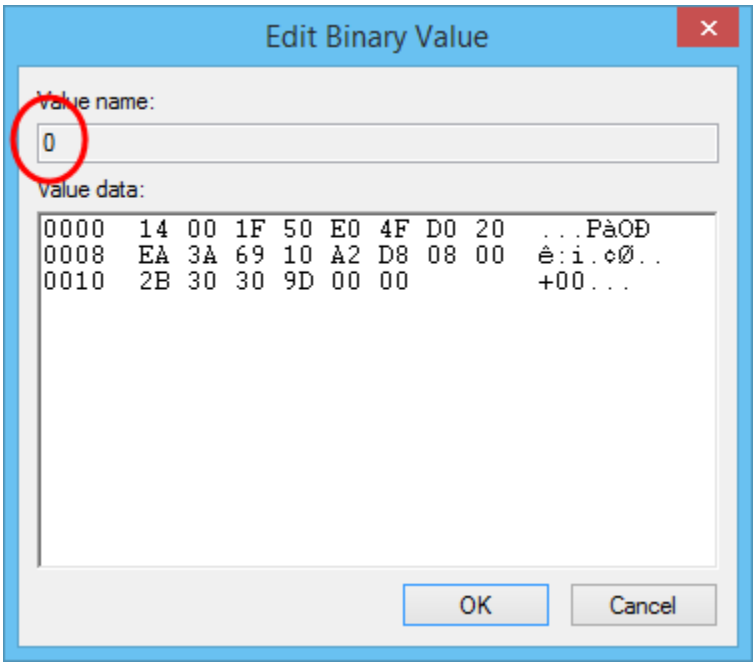


On the left, registry keys are displayed. After selecting a key, the values for that key are displayed in the right pane.

The values on the right correspond to ShellBags that are children of the selected key. In the screen shot above, it is the BagMRU key. The BagMRU key represents the “Desktop” in Windows and all ShellBags will exist under the Desktop. In other words, the Desktop is the topmost Shellbag.

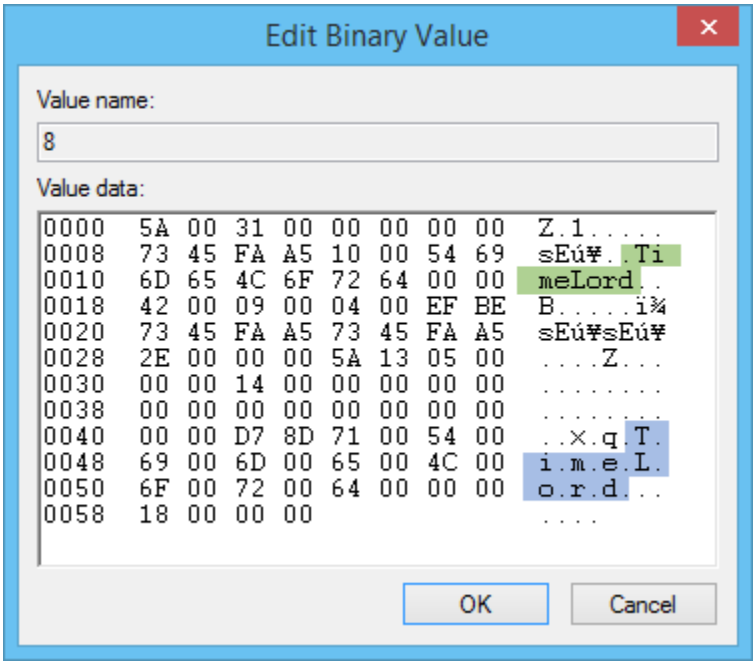
ShellBag binary contents

Each ShellBag contains binary data that defines what the ShellBag represents. Double clicking on a value will bring up an editor showing the binary data as seen below.



In the example above, the “0” value has been opened. The contents of the ShellBag are shown in hexadecimal. The example above contains a GUID which corresponds to the “My Computer” folder. There are hundreds of GUIDs that map to directories, control panel items and categories, etc. SBE contains hundreds of GUID mappings to human readable formats.

Some ShellBags will contain strings (both ANSI and Unicode) representing things such as directory names or UNC paths. In the image below, a directory named “Timelord” was accessed. The ‘short name’ is highlighted in green and the ‘long name’ is highlighted in blue.

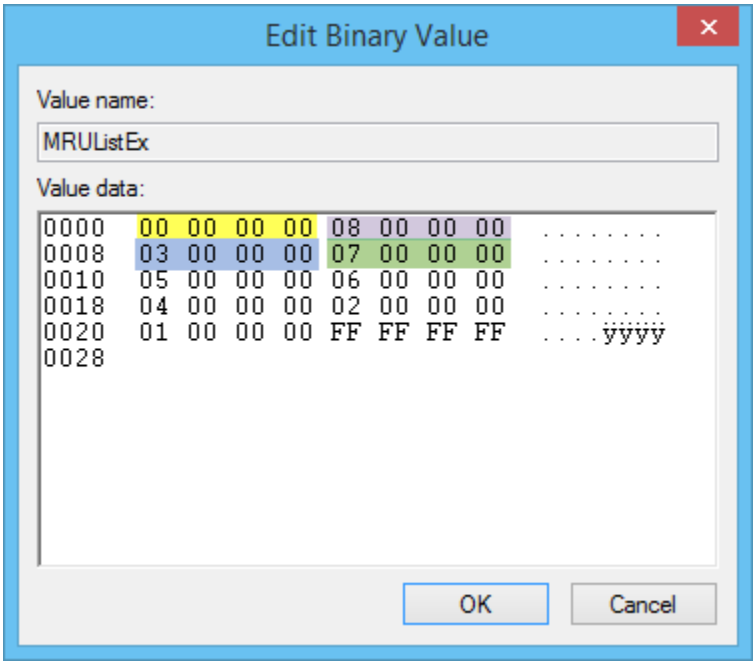


While not obvious, there are also several dates and times embedded in the binary data as well as MFT entry and sequence numbers. It is also possible to determine the file system this directory existed on based on the contents of the ShellBag. In the example above, the directory existed on an NTFS file system with MFT Entry Number 332634 and MFT Sequence Number 20 and was last accessed on 11/19/2014 at 8:47:52 PM +00:00!

MRUListEx

There is also a value named MRUListEx. This is the Most Recently Used list and reflects the order the ShellBags were opened with the most recently opened bag being listed first. As ShellBags are opened, the MRUListEx values are shifted to the right and the most recently opened value is added to the leftmost position.

Double clicking the MRUListEx value brings up an editor showing the binary data contained in the value.



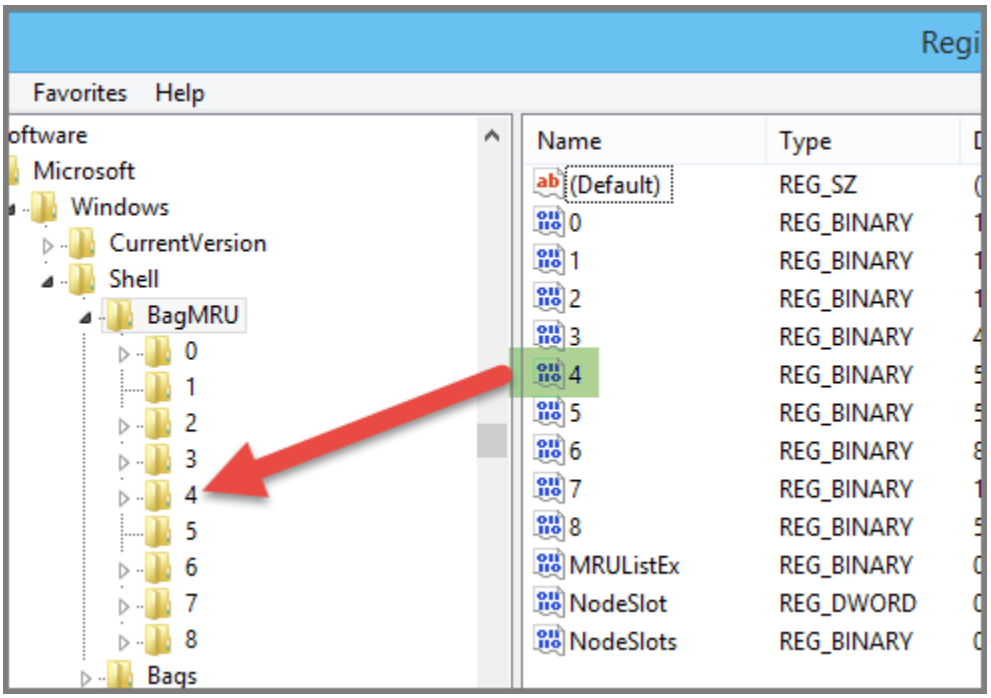
Each entry in the MRU list is 4 bytes long (little endian). In the image above, the first 4 MRU positions have been highlighted with different colors.

Based on what is in the MRUListEx key above, the order the ShellBags under BagMRU were opened is 0, 8, 3, 7, 5, 6, 4, 2, 1. The last entry in MRUListEx is always FF FF FF FF.

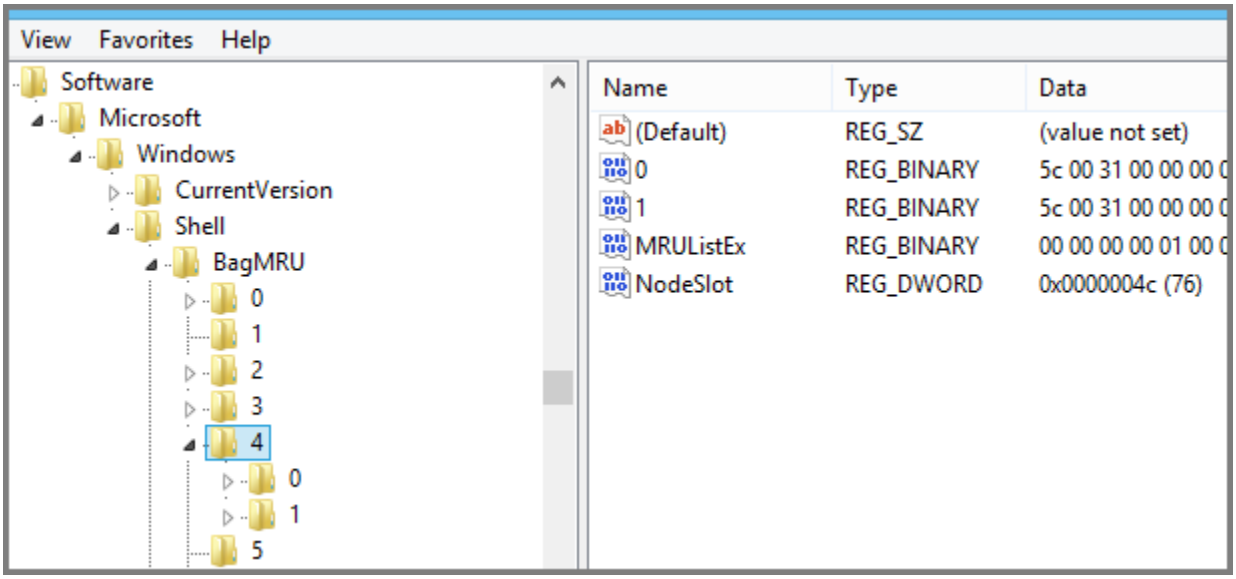
Subkeys and their relation to values

In the screen shot above, the BagMRU key has been expanded and circled in red. Under it are subkeys 0-8. Notice there are also values with the same name on the right.

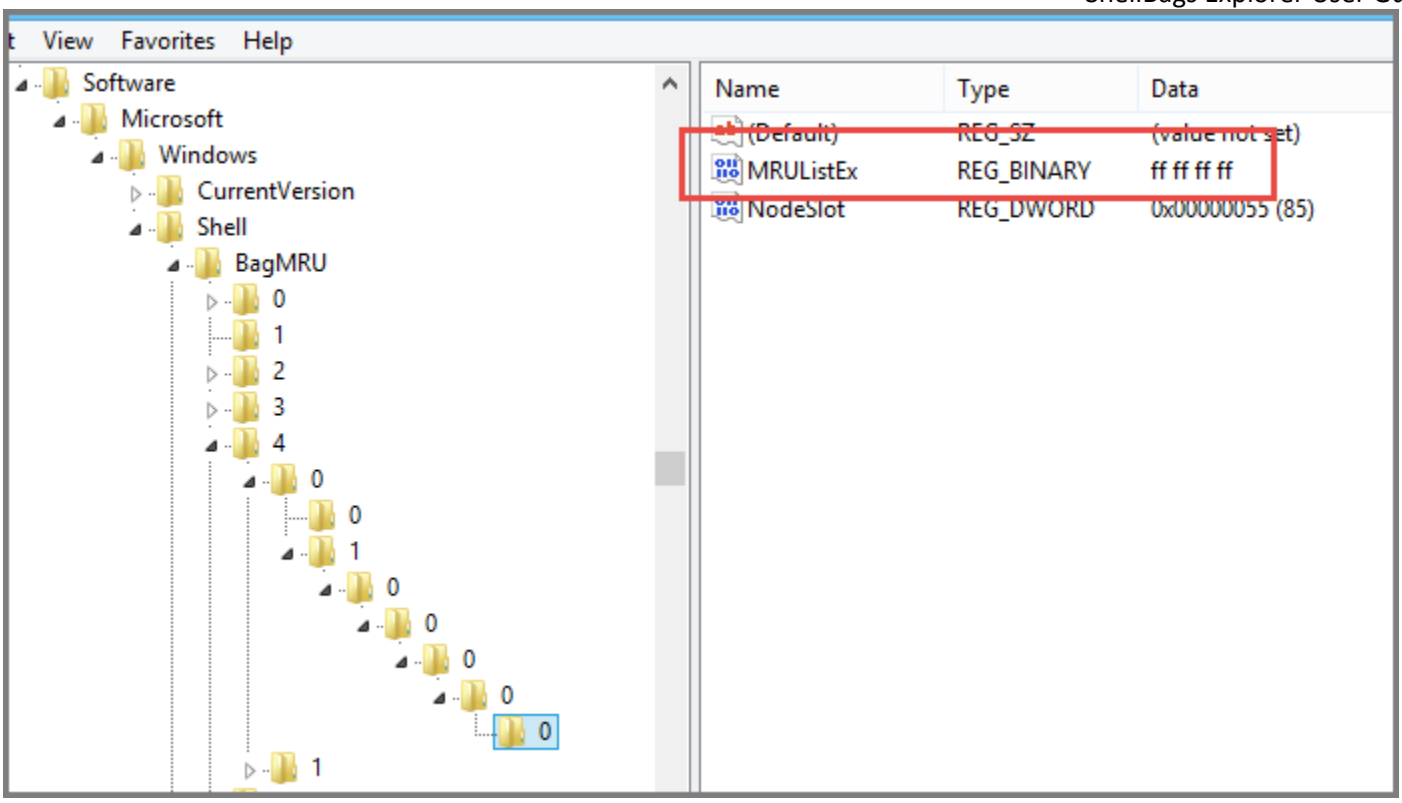
Subkeys with the same name as the value will contain child objects for that value. Recall each value represents a ShellBag. A ShellBag can be the Desktop item, a Control Panel Category, a Control Panel item, a drive letter, or a directory (there are other possibilities as well).



In the image above, the ShellBag with a value of 4 is the "parent" ShellBag for any ShellBags found in the key with the same name. Selecting the key with name 4 results in the following.



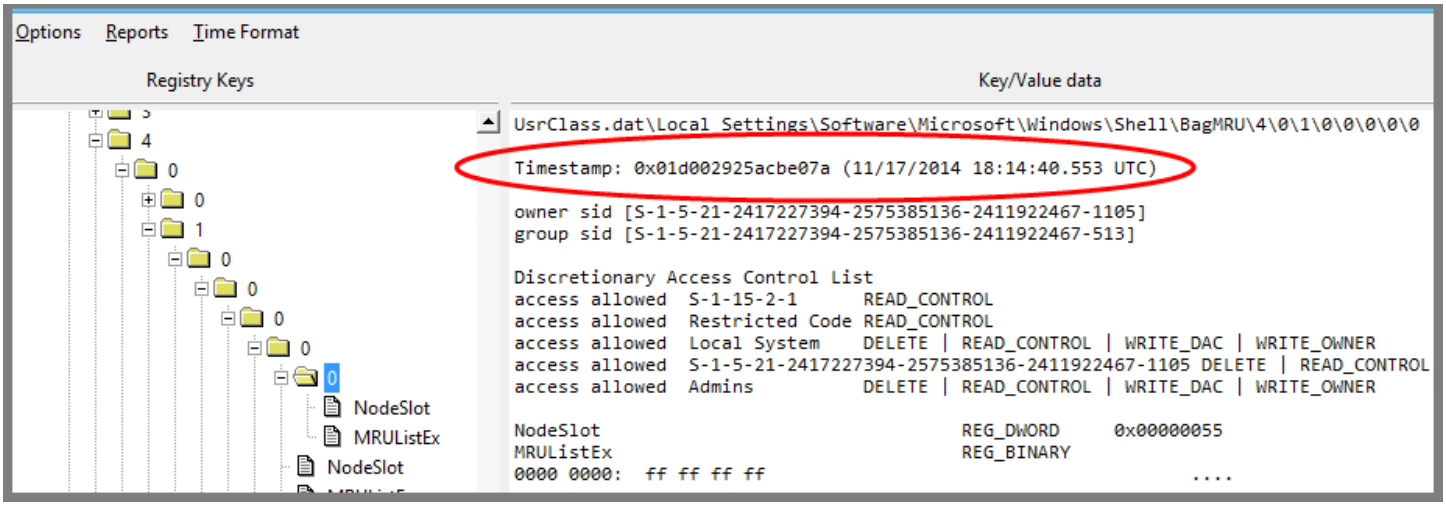
As you can see, the pattern continues as we drill down into child keys. If we continue to drill down we will eventually run out of child keys as seen in the next screen shot.



We know we are at the "bottom" because there are no more child keys and the MRUListEx value is FF FF FF FF.

LastWrite times and their value in ShellBags analysis

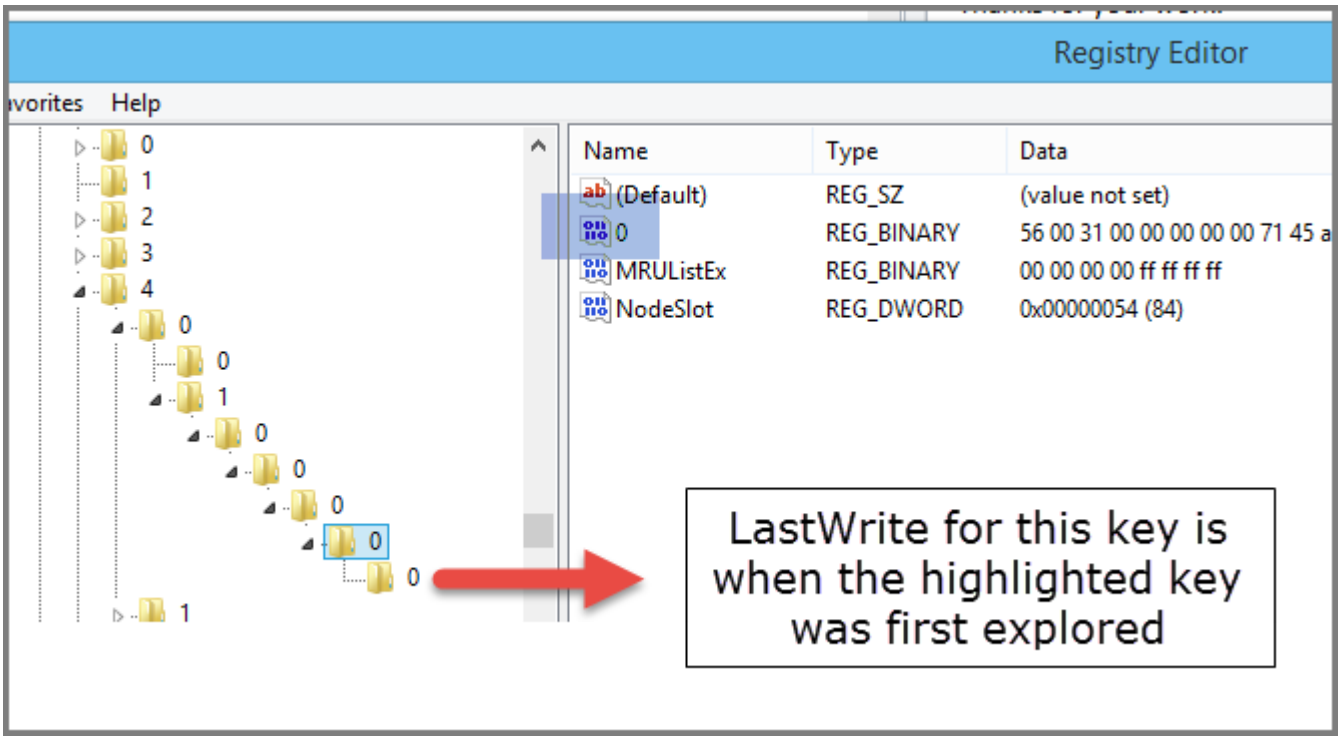
Whenever a value is changed in a registry key, the key's LastWrite timestamp is updated. Regedit does not display these timestamps, but other forensic programs such as X-Ways Forensics or TZWorks yaru can. When using yaru to look at the same key as shown above, we can see the key was last written on 11/17/2014 at 18:14:40.553 UTC.



Using LastWrite values to determine First Explored and Last Explored dates and times

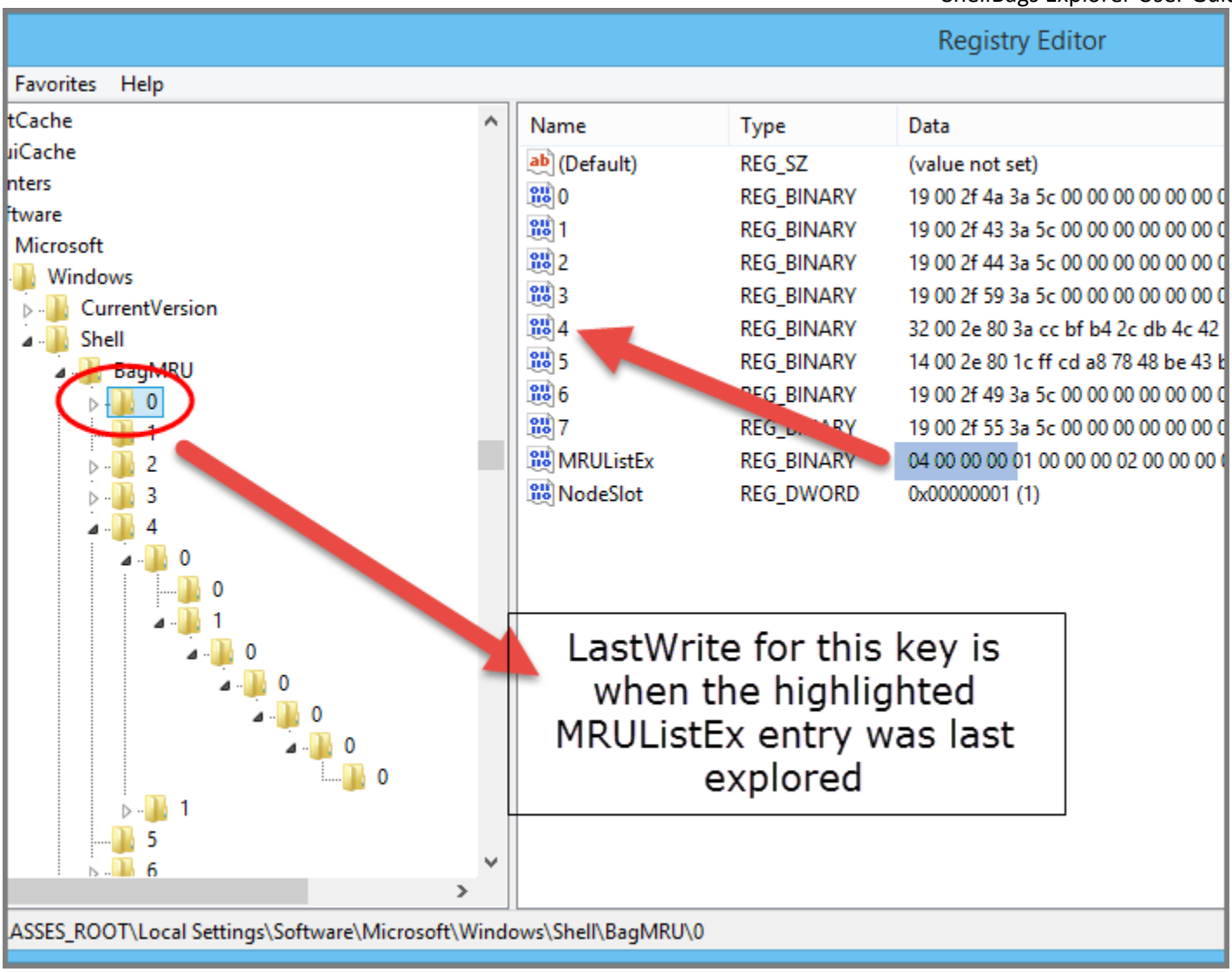
Keeping in mind how LastWrite timestamps work, we can now determine when a given ShellBag was accessed.

To determine the first time a ShellBag was explored, we can look for the “bottom” most keys and use their LastWrite timestamps for the ShellBag in the parent key. Recall in the example above, the key is “0” and the last write value is 11/17/2014 at 18:14:40.553 UTC. If we go “up” a level and select the parent key of the bottommost key, we see a ShellBag value named “0.”



Note: This only works for the bottommost key and its corresponding value in the key’s parent. If a child directory is browsed underneath the highlighted key above, it is not possible to determine the first explored date for the intermediate folder anymore (but we would now know when the newly browsed folder was viewed, assuming it was the bottommost directory navigated to).

To determine the last time a ShellBag was explored, we use the LastWrite timestamp in conjunction with the MRUListEx value. Recall the first entry in MRUListEx is the most recently viewed ShellBag in a given key. By looking at the LastWrite timestamp for a key, we know when the first entry in the MRUListEx was last explored.



Since MRUListEx has its first entry set to a value of "4", the ShellBag with the same value was last explored when the "0" key was last written to. Since the MRUListEx value is updated as ShellBags are accessed, the LastWrite timestamp is also updated.

More information on these concepts is available at http://www.4n6k.com/2013/12/shellbags-forensics-addressing.html.

Why another ShellBags program?

Tools like RegRipper and sbags from TZWorks have processed ShellBags for quite some time now, but ShellBags Explorer (SBE) is different in that it presents a visual representation of what a user's directory structure looked like. Additionally SBE exposes various timestamps such as First Explored and Last Explored for a given folder, the file system where a directory existed, and so on. SBE also contains a command line version which can produce output (while still including the additional relevant timestamps and file system info, etc.) similar to sbags and RegRipper.

Capabilities overview

SBE is meant to be an all-inclusive tool for ShellBag artifacts. It negates the need for laborious manual steps, decoding of data, and determining contextual relationships between directories, etc.

- Included support for all known Extension blocks and auto-detection of unknown blocks, unknown ShellBag types, etc.
- Data interpreter to Hex view. As hex values are selected, the values update from the cursor's position.
- Support for NTUser.dat and USRClass.dat
- Consistent display of data for bags
- Ability to view all bags recursively to easily sort, filter, etc.
- Ability to ingest multiple registry hives and remove duplicate ShellBags. This allows for a comprehensive view of directory access spanning the range of data in all registry hives.
- Ability to show what directories were accessed on CD and DVD media (and therefore showing what drive letters were optical readers)
- And MUCH more

How does it work?

- The basis for SBE is Joachim Metz's document, Windows Shell Item format specification, which is available here http://goo.gl/rJXmLY. The specification outlines all known shell bag types and layouts and is continually updated by Metz as new information is discovered.
- Certain shell bags contain other properties with complex layouts. These specifications are linked in the document above.

SBE has been designed with features and capabilities such as unknown GUID detection, extension block mismatches, etc. and has been used to contribute significant, previously unknown information to the format specification for shell items. By automatically reporting anomalies in the decoding process, these anomalies can be reported to the program developer in order to improve the capabilities of SBE and thusly the forensic process in which SBE is used.

Why use ShellBags Explorer and not ShellBag (sbag) Parser by TZWorks, MiTeC Registry Analyser or RegRipper?

Each of the three other tools are summary tools and none show you the contents of all (or nearly all) bytes in ShellBags. In fact, some of them do not show certain bags at all. SBE parses hives that crash other tools and report far more data than others as well. Additionally, only SBE (outside of a hex editor) can truly be used to study what is inside ShellBags while at the same time representing data in such a way that is visually similar to how the end user would have seen their file systems. By representing ShellBags in a hierarchical fashion vs. a flat list, an examiner can more easily explore and see directory structure and layout as the subject did. Additionally, SBE allows for filtering, sorting and grouping far beyond the capabilities of Excel or similar tools.

Getting started

If you are reading this, it is assumed you already have ShellBags Explorer installed somewhere on your machine. A default installation of ShellBags Explorer includes the following files:

- SBECmd.exe: The command line version of ShellBags Explorer

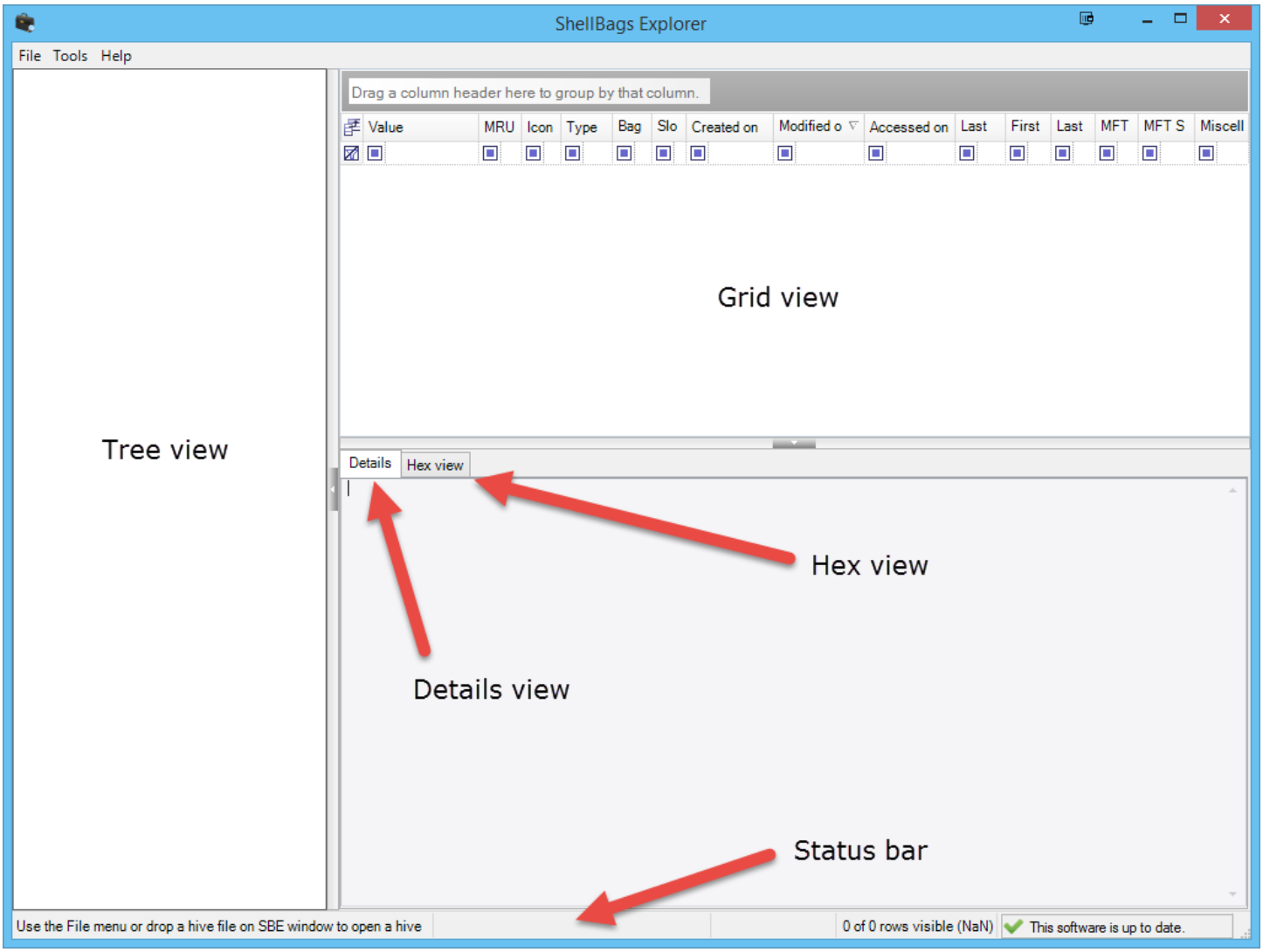
- ShellBagsExplorer.exe: The GUI version of ShellBags Explorer
- ShellBagsExplorerManual.pdf: The file you are reading now

Either version of ShellBags Explorer (SBE) can be used to process and examine registry hives. One program does not rely on the other for SBE to function.

ShellBagsExplorer.exe

This is the GUI version of SBE. It provides deep insight into shellbag data in an easy to use interface that resembles Windows Explorer.

To start SBE, open the SBE folder and double click on ShellBagsExplorer.exe.



Each of the sections above will be explored further in subsequent sections below.

Tree view

The tree view displays a Windows Explorer like representation of ShellBag data. There are menu options under Tools to automatically expand and contract all nodes.

Left clicking a tree node

Left clicking a node will display details of the selected node in the Details view pane. Additionally, all child bags of the selected node are displayed in the grid view.

Right clicking a tree node

Right clicking a node in the tree will recursively load all child nodes of the selected node. The nodes are displayed in the grid view.

Grid view

The grid view displays all ShellBags located below the selected node in the tree, or, in the case of recursive viewing, a list of all ShellBags located below the selected node and all child nodes of those nodes.

Left clicking a grid entry

Left clicking a node in the grid view will display the details of the ShellBag in the Detail view. The hex view will also be updated to reflect the binary contents of the ShellBag.

Right clicking a grid entry

Right clicking a node in the grid view will select that node in the tree view. This is handy when viewing nodes recursively to see where in the directory structure a particular node lives.

Details view

After a ShellBag is selected, the details view pane displays all known data about the selected ShellBag. This includes basic information as well as extension blocks, MFT information, file system hints, raw hex content, and so on.

Hex view

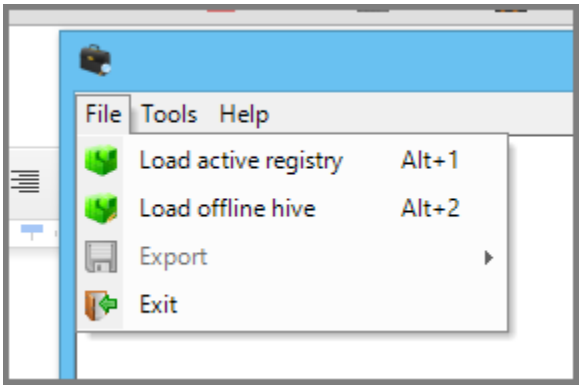
After a ShellBag is selected, the hex view will contain the raw hex content of the selected ShellBag. To the right of the hex display is a data interpreter that is updated in real time as data is selected in the hex editor. This allows for verification of data in both the grid and details view.

Status bar

The status bar on the bottom contains various details about the loaded hive, whether or not a filter is active (including the number of visible rows vs. the total) and if there are any updates available to SBE.

Menus

File



The File menu allows for loading either the live registry information or an offline hive. When loading an offline hive, the user will be prompted for the location of the hive file to load.

More information is available when browsing an offline hive than browsing the live registry. This is because as of SBE version 0.5.0.0, the LastWrite timestamp for registry keys is not available when looking at the live registry. The live registry option can be used to explore ShellBags from the live machine during training, research, and so on.

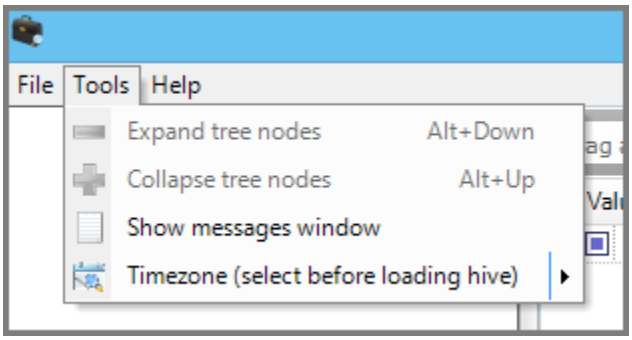
Export submenu

Once SBE has loaded a hive, the File menu also allows for exporting of the data in one of several formats.

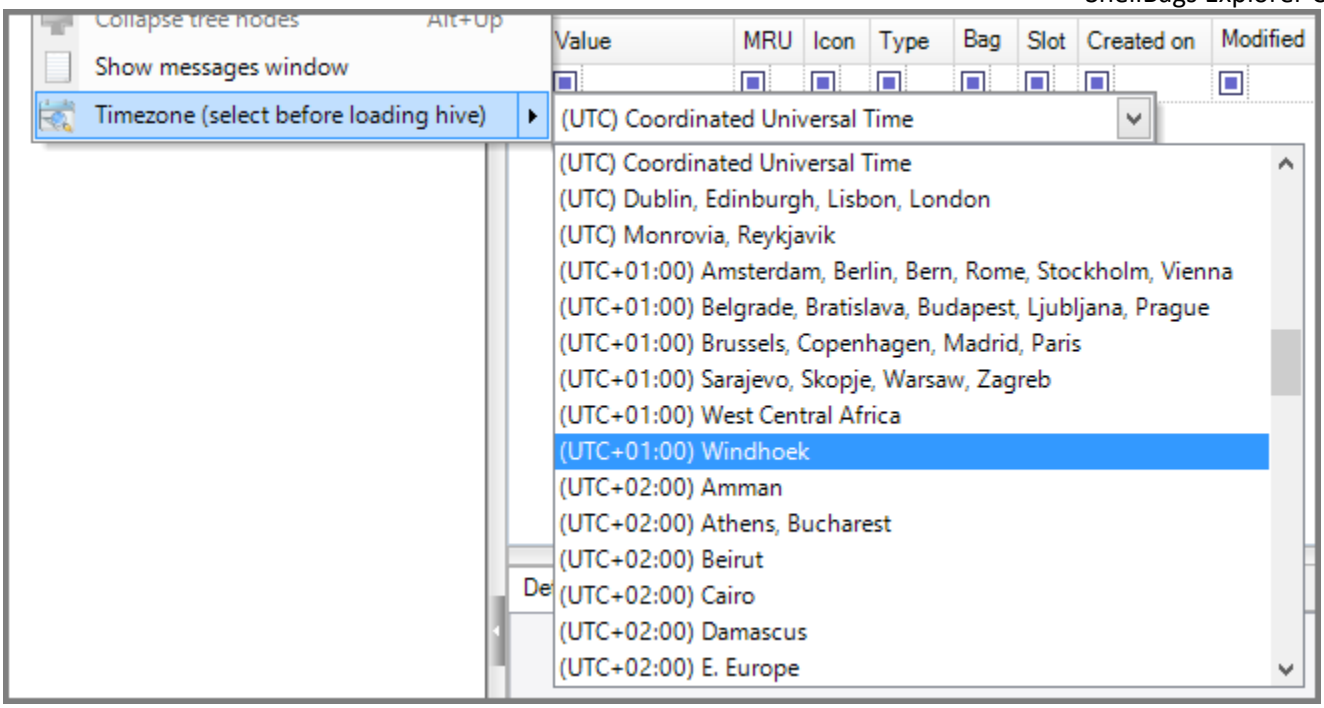
- TSV: All available columns will be exported
- Excel: The visible columns in the Grid view are exported.
- json: All data is exported in json format.

NOTE: You can customize the columns shown in the grid view by clicking the field chooser icon in the upper left corner of the grid (this is explained in more detail in an upcoming section). This allows you to hide or show whatever columns you choose to make your work easier.

Tools

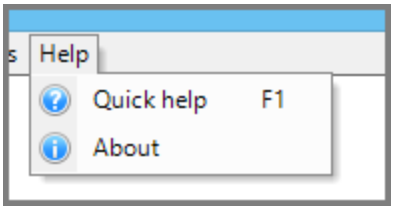


The tools menu contains options to expand and contract tree nodes, show the messages window, or set the time zone to be used when displaying dates and times. The messages window is displayed automatically as messages are generated.



A time zone must be selected before selecting a source for ShellBags (the live registry or an offline hive). By default, UTC is used for all dates and times.

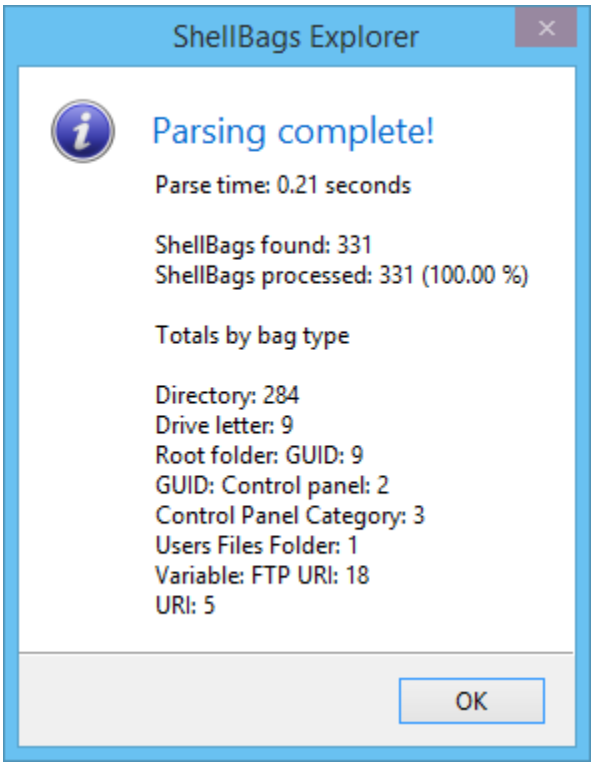
Help



Quick help provides basic information on how to use SBE. About contains version information and contact information for the developer.

Workflow overview

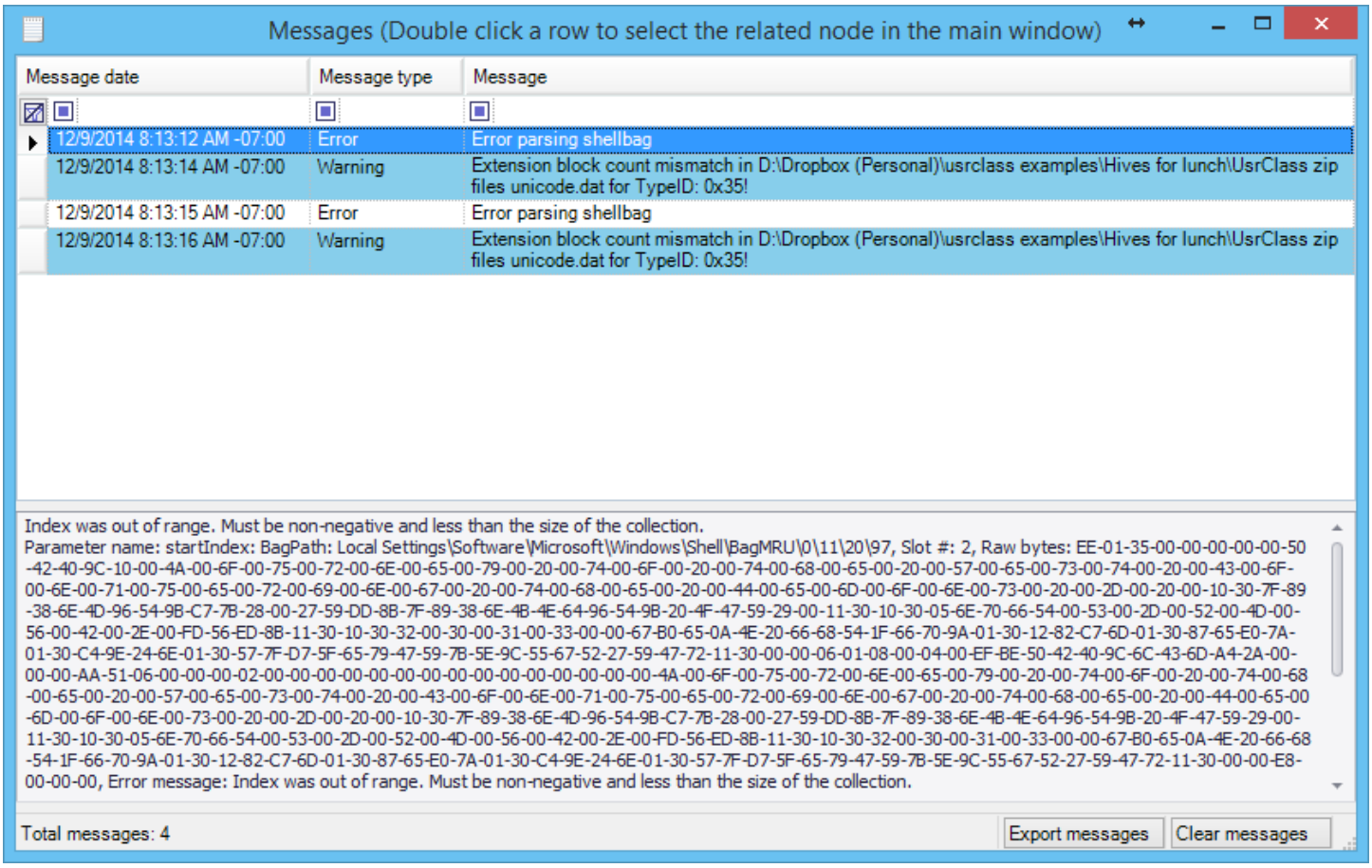
After selecting either the live registry or an offline hive, SBE will display a summary of the types of ShellBags that were found as seen below.



In the image above, notice 331 ShellBags were found in the registry and 331 ShellBags were processed. Should there be a situation where the number of ShellBags found is not equal to the number of ShellBags displayed by SBE, the summary screen will be different to reflect this fact.

Messages window

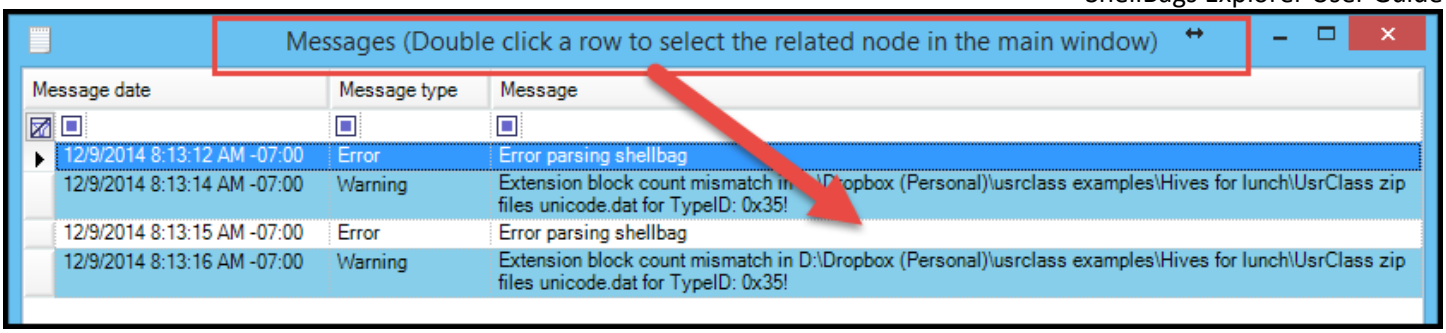
If there are any errors when processing ShellBags, the Messages window will be displayed.



Selecting a message will show the details for that message in the lower pane. All messages can be exported or cleared using the buttons in the lower right corner.

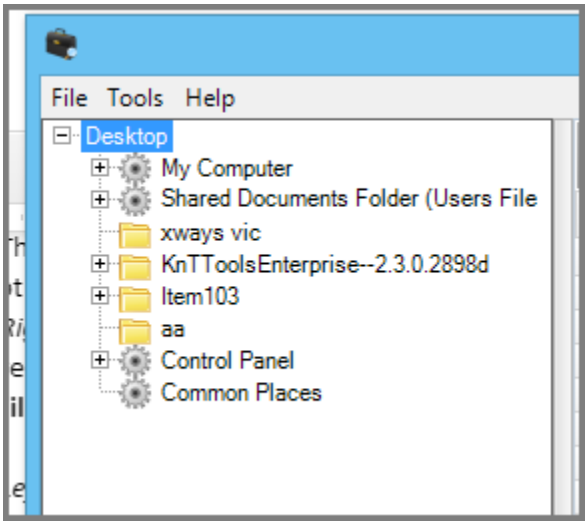
The Messages window can contain both errors and informational messages. Informational messages will be displayed when unknown GUIDs are found, new extension blocks are discovered, or there is a mismatch between the number of extension blocks in a ShellBag and the number of parsed extension blocks (these concepts will be elaborated on soon).

NOTE: You can quickly jump to the ShellBag related to the message by double clicking anywhere on a row. This will select the ShellBag in the tree so it can be reviewed as needed.

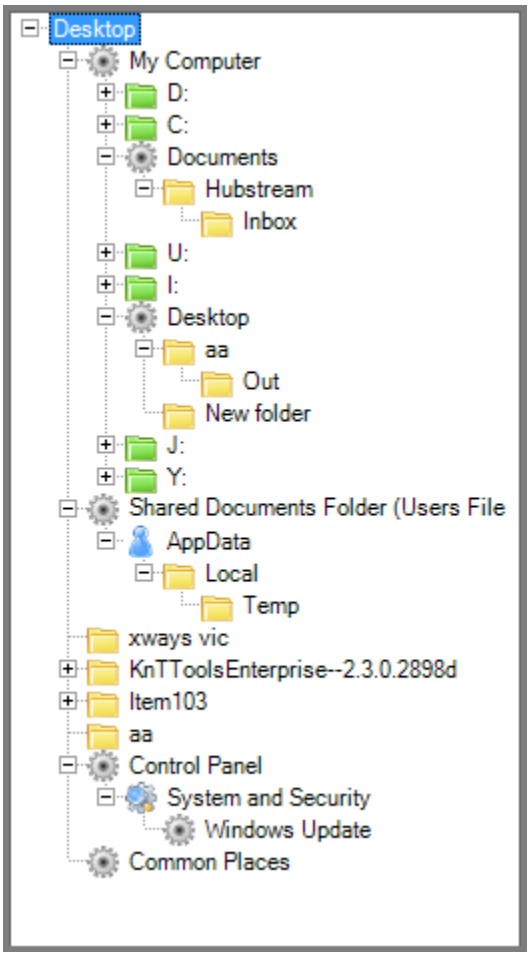


Tree view

The tree view will also be updated. As seen earlier, the starting point for ShellBags is the Desktop folder.

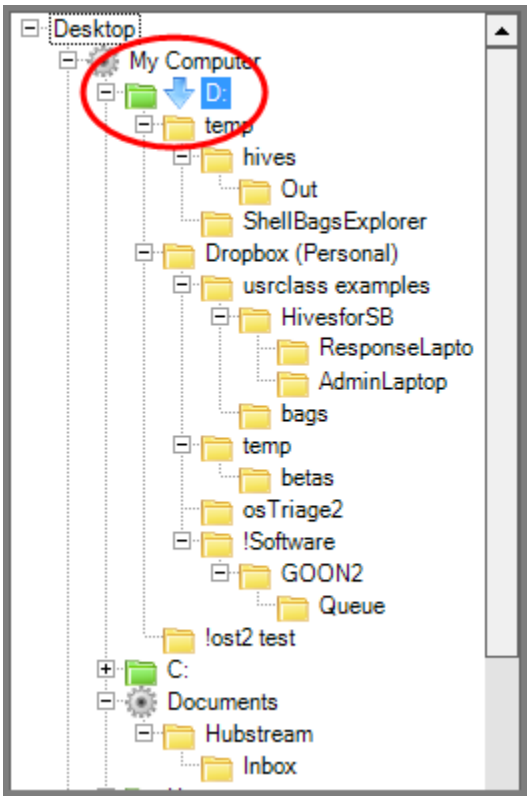


Expanding a node displays its children. In the image below, several child objects have been expanded.



Each different kind of ShellBag is represented by a different icon. In the example above we see icons for directories, GUIDs, control panel categories, drive letters, and user application data. This helps you quickly visualize what kind of ShellBags are available.

As mentioned above, left clicking selects a given node in the tree and displays child bags in the grid view. Right clicking a node will select that node and expand all child nodes and display all bags under the selected node in the grid view.

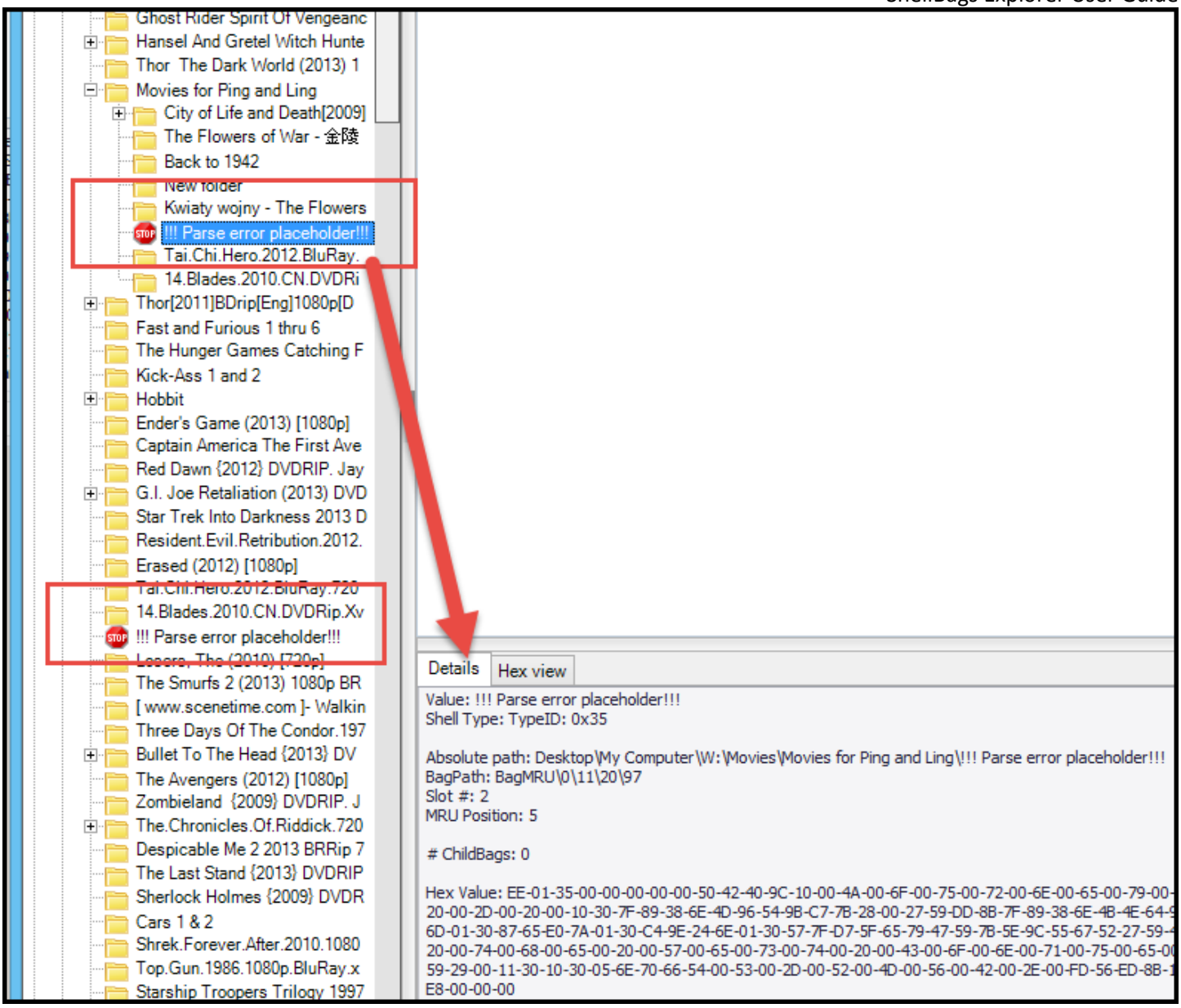


In the example above, D: has been recursively explored. Notice the icon now shows a blue down arrow indicating the node is being explored recursively. Exploring recursively lets you drill down into specific areas of interest and, as we will soon see, easily search for things across a given set of ShellBags.

ShellBag parsing errors

Should any errors occur when parsing ShellBags, a placeholder for the ShellBag will be added as shown in the image below. Some details for the unparsed ShellBag will be displayed in the Details pane. The Hex view allows for reviewing of these ShellBags for relevant data and a message is added to the Messages window when this occurs.

By adding a placeholder for ShellBags with parse errors, any child ShellBags of the unparsed ShellBag will be still be visible in the proper context.



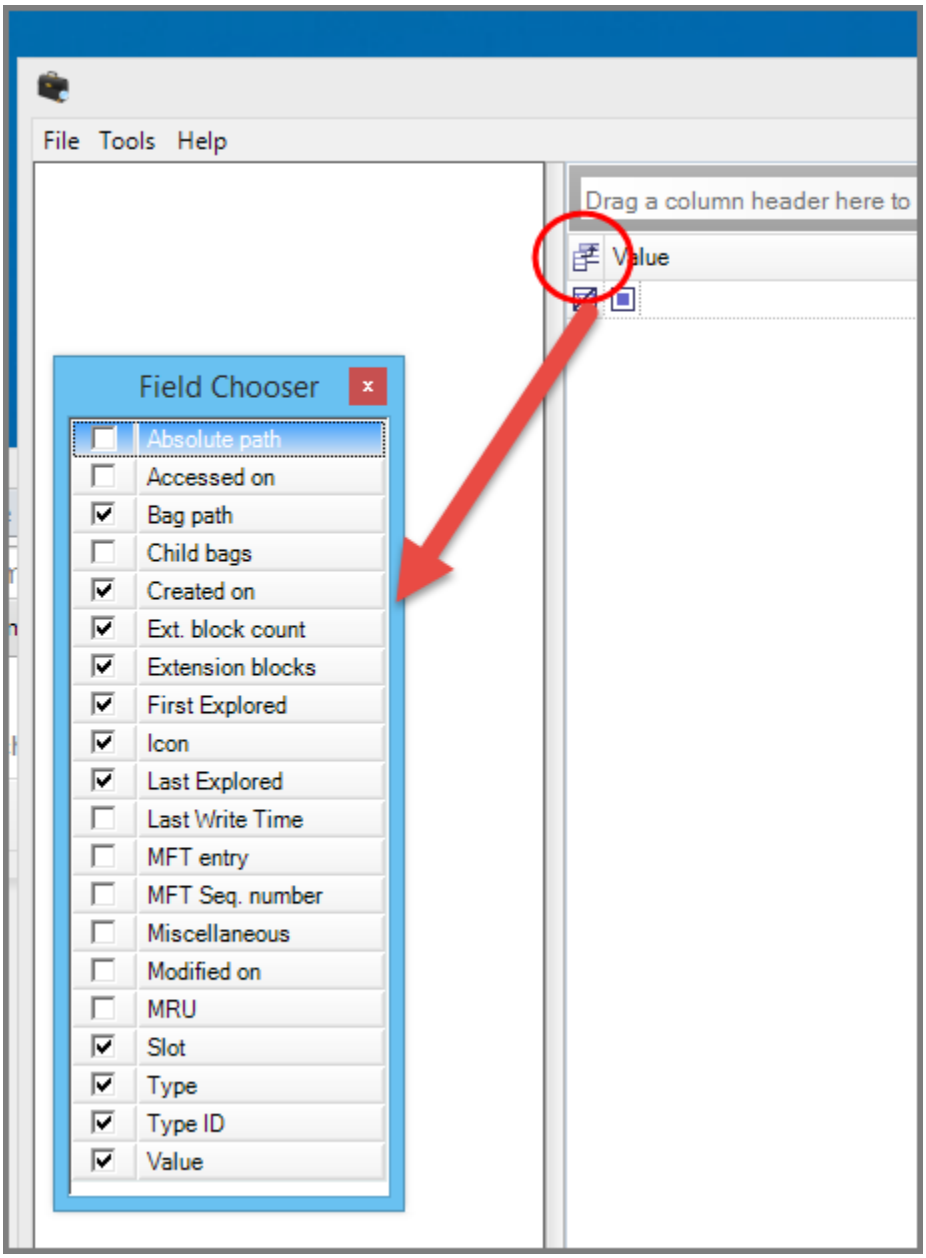
Grid view

As nodes are selected in the tree view, the grid view is updated to show the child ShellBags of the selected node. In the case of recursive viewing, all child bags from the selected node down are displayed in the grid view.

The grid view serves as a way to view Shellbags in a consistent manner. Not all bags have data for all of the columns, but for the most part the columns available in the grid view provide a homogeneous view of ShellBags. Because of this, common features of different bag types can be grouped on, filtered, sorted by, etc.

Customizing grid view columns

Clicking in the upper left corner of the grid view allows for customizing the fields displayed in the grid.



Column definitions

SBE supports the following columns as of v0.5.0.0:

- Absolute path: The absolute path from the Desktop to the location of the shell bag
- Accessed on: The access date as stored in the ShellBag
- Bag path: The path to the ShellBag relative to the BagMRU key
- Child Bags: Number of child bags of the parent bag key
- Created on: The created date as stored in the ShellBag (usually based on MFT data)
- Ext. block count: The total number of extension blocks found in this shell bag
- Extension blocks: The names of unique extension blocks found in the shell bag
- First Explored: When available, the timestamp a folder was first explored
- Icon: A visual identifier for the shell bag

- Last Explored: When available, the timestamp a folder was last explored
- Last Write Time: Only available when loading an offline hive, this is the timestamp the ShellBag KEY was last updated.
- MFT entry
- MFT Seq. number
- Miscellaneous: Used to report additional items of interest about shell bags as needed. This is primarily used to report the type of file system an item was located on
- Modified on: The modified date as stored in the ShellBag
- MRU: The Most Recently Used position of the bag
- Slot: The shell bag's numerical identifier in its parent key
- Type: The human readable description of what kind of data the shell bag represents (file, folder, etc.)
- Type ID: The hexadecimal identifier for the shell bag. This is found in offset 0x02 in the hex that makes up a shell bag.
- Value: The name of the file, folder, property view, etc. for the shell bag. This is the primary identifier for a shell bag.

Note: The MFT entry and sequence number can be used to determine the file system a particular bag came from:

- If entry number > 0 and sequence number > 0, then the file system is NTFS.
- If entry number > 0 and sequence number == 0, then the file system is FAT. (Further checks against the accuracy of Last Access is then done to determine FAT vs. exFAT)

When applicable, the Miscellaneous column will indicate which file system the ShellBag target originated from. Thanks to David Cowen for this idea and for testing.

Changing column order

Column order can be adjusted by dragging and dropping columns to new positions in the grid. Any changes are persisted.

Interacting with the grid

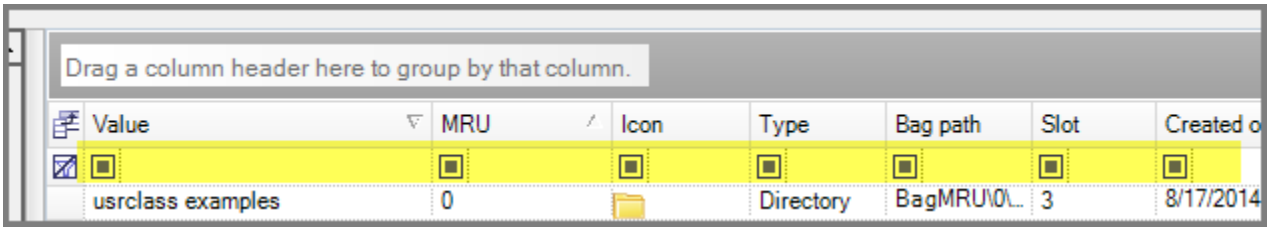
The grid view can be used to sort, filter, and group ShellBags.

Sorting

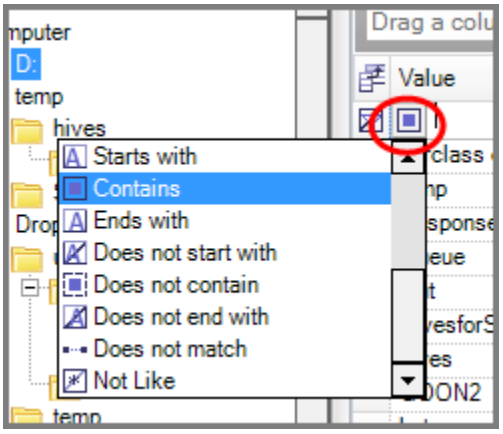
To sort, simply click a column header. Click it again to sort in reverse order. To sort by more than one column, click the initial column to sort by, then hold SHIFT and click on another column. The second column will be sorted while the first column retains its initial sorting.

Filtering

At the top of each column is a filter cell. In the image below it is highlighted in yellow.

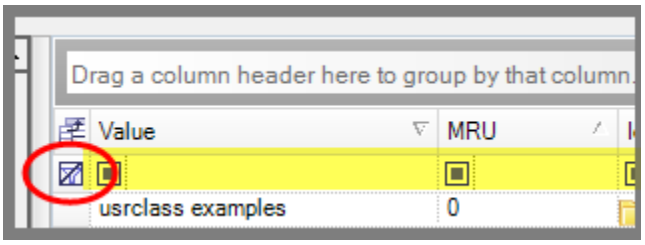


Clicking on this cell and typing will immediately filter that column to only items containing the entered text. To the left of the cell is a button with options on how the filter should work. The default is “contains.”



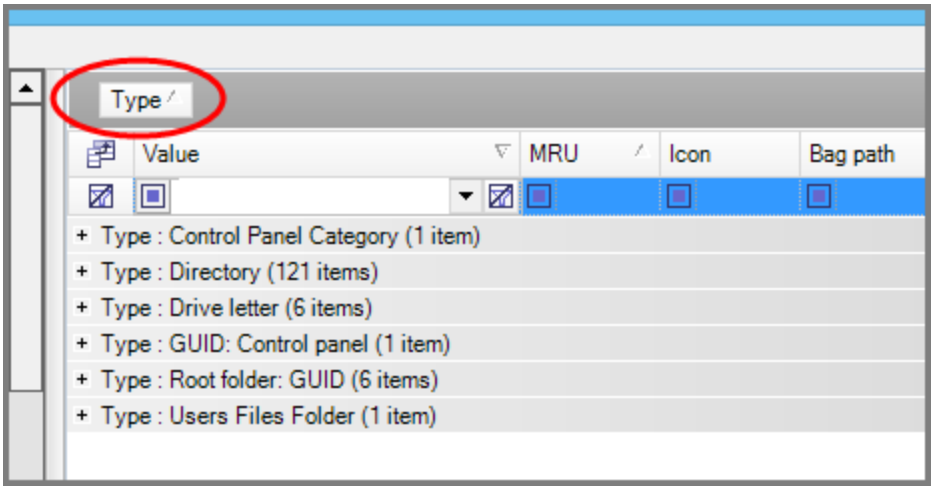
There are also dropdowns which contain all unique entries in the column plus options to add custom filters to the list. To the right of each cell is a button that can be used to clear that column’s filter.

To the far left of the filter row is a button that can be used to clear all filters.

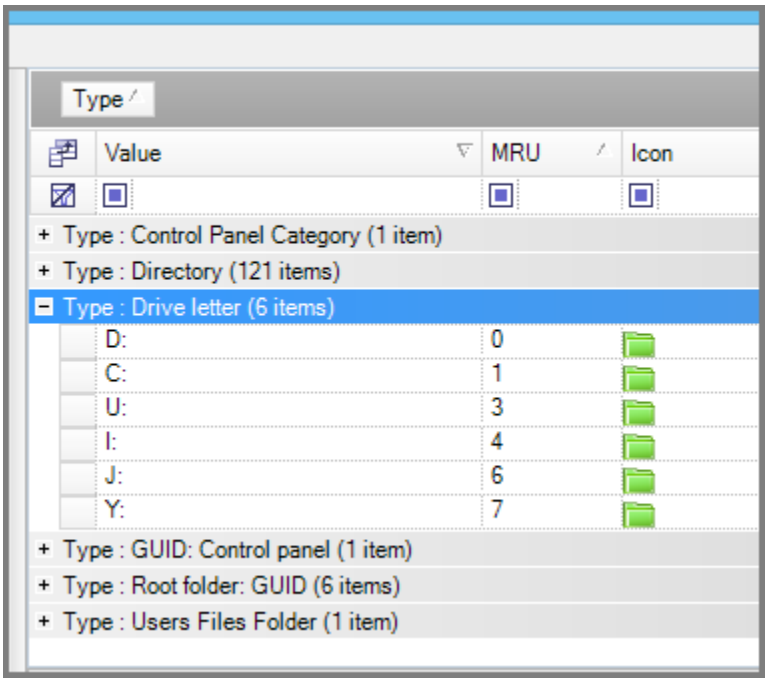


Grouping

At the top of the grid is an area where one or more column headers can be dragged. As columns are dragged and dropped, each unique value in the selected column will be used to create a group.



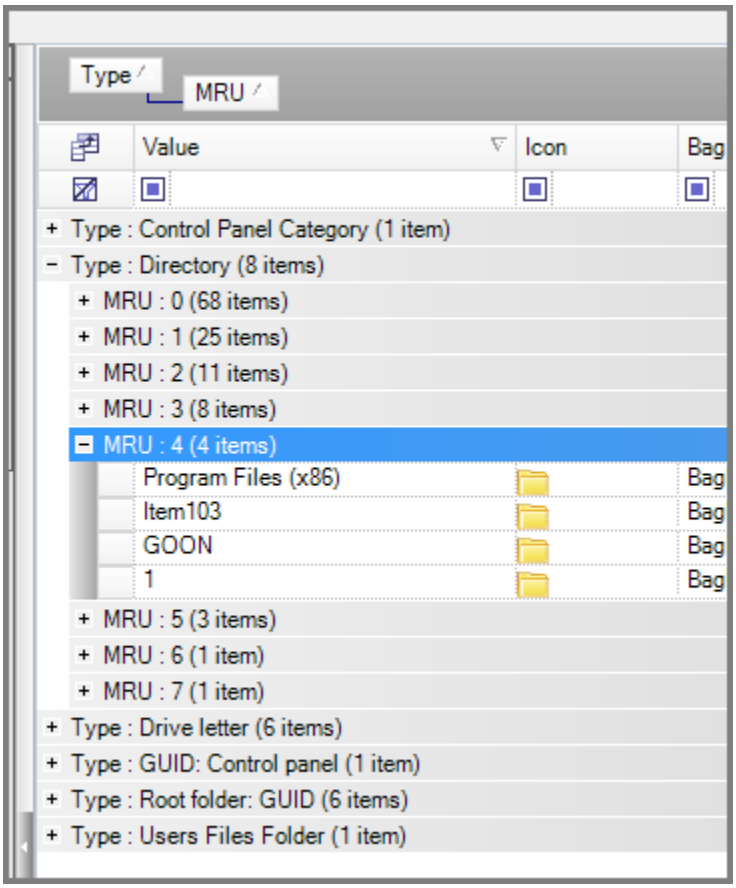
In the example above, the “Type” column was grouped. Clicking on a plus sign expands that group.



Clicking the minus sign collapses it.

After grouping, columns can be sorted as we saw earlier, by clicking on the column header.

To group more than one column, drag them to the group area.



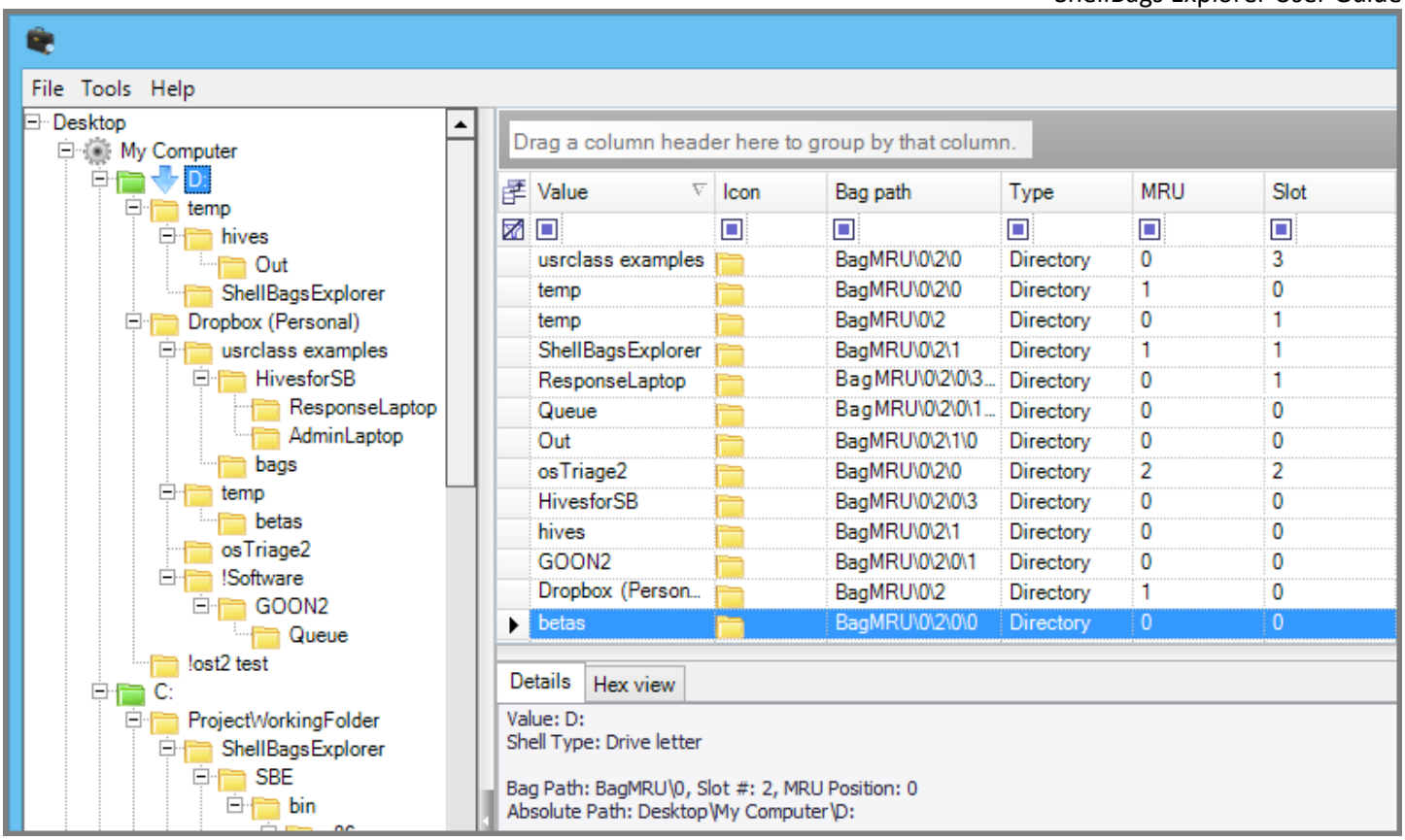
In the example above, we first grouped by type, then by MRU. This can be useful to see a list of all the most recently accessed directories (i.e. all the ShellBags of type directory where MRU == 0) .

To “undo” grouping, drag the columns back into the main grid area at any location.

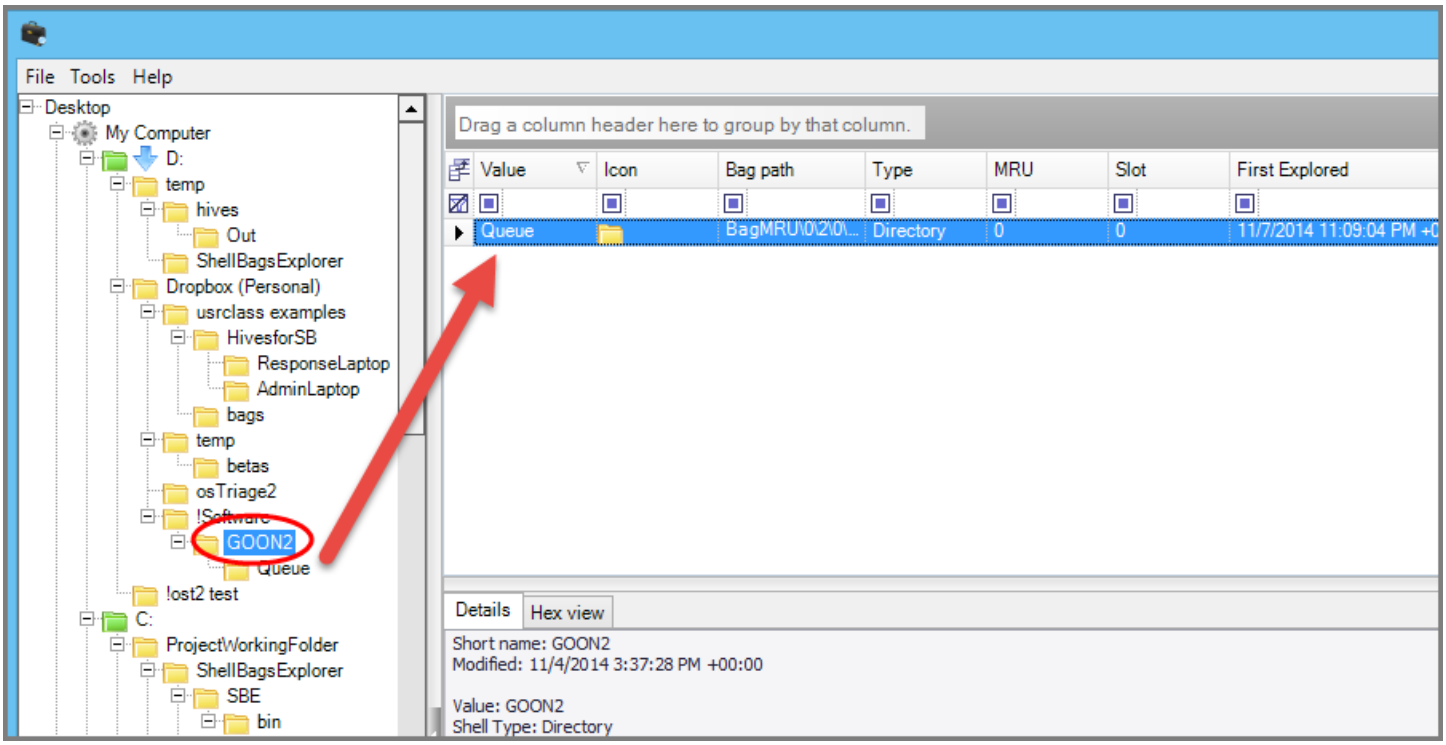
Selecting bags in grid view

Left clicking a bag will select that bag and display that bag’s details in the details view (described below). After selecting a bag via left clicking it, you can jump to that bag by right clicking it.

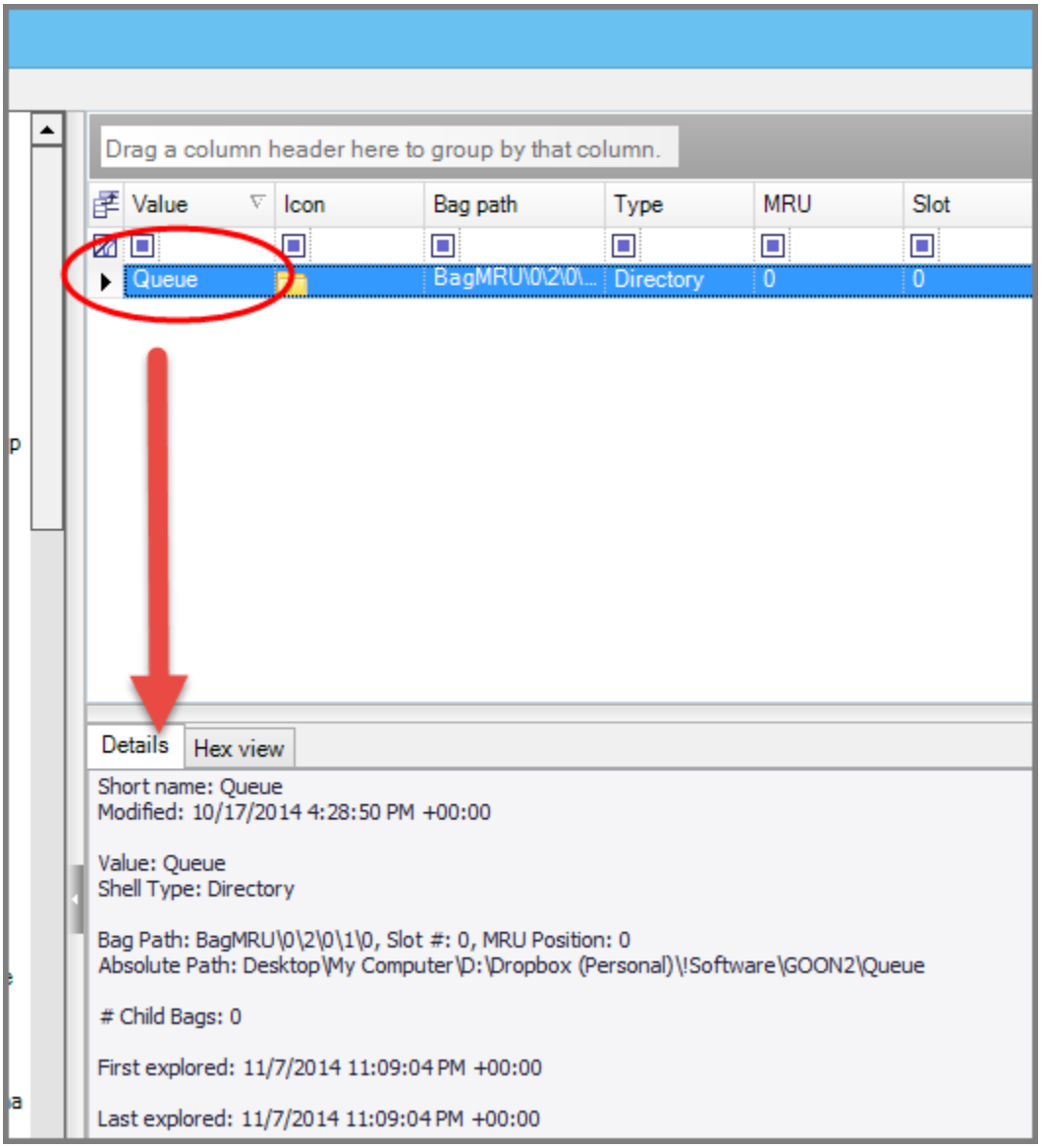
In the image below, we are recursively viewing the bags at and below D:.



Notice the D: drive is highlighted in the tree view and the details view reflects this. If the bag with value "GOON2" is left clicked, then right clicked, notice the tree view is updated and the "GOON2" bag is highlighted and the grid view is updated to show the child bags of the GOON2 bag.



The details pane is also updated with the newly selected bag in the tree view. Clicking the “Queue” bag in grid view will update the details view to the “Queue” bag.



Details view

The details view contains all known ShellBag information in human readable (and hopefully consumable) format. Binary data is converted to timestamps, strings, numbers, and other structures.

ShellBags in general have the following structure (For an exhaustive breakdown of ShellBag binary layouts, including extension blocks, see the Metz document referenced above (http://goo.gl/rJXmLY)):

- Offset 0-1: Size of the ShellBag
- Offset 2: Type indicator
- Type specific data

The type specific data contains things such as timestamps, file attributes, strings, and extension blocks. Extension blocks contain additional data relevant to a ShellBag which can be used in different ShellBags. Because of this, extension blocks have their own formatting that is consistent across bags.

Extension blocks can be identified by a unique signature, BE EF 00 XX, where XX varies on the type of extension block. In the image below, the beginning of a BEEF0004 block is highlighted. Since the data is stored in little endian format it has to be read “backwards.”

```
00 01 02 03 04 05 06 07 08 09 0A 0B
60 00 31 00 00 00 00 00 43 45 F1 72   `.1.....CEñr
10 00 41 44 4D 49 4E 4C 7E 31 00 00   ..ADMINL~1..
48 00 09 00 04 00 EF BE 43 45 EC 72   H.....ï¾CEìr
45 45 74 A0 2E 00 00 00 2C 00 00 00   EEt .....,...
00 00 3B 00 00 00 00 00 00 00 00 00   ..;.........
00 00 00 00 00 00 88 A2 18 00 41 00   .......¢..A.
64 00 6D 00 69 00 6E 00 4C 00 61 00   d.m.i.n.L.a.
70 00 74 00 6F 00 70 00 00 00 18 00   p.t.o.p.....
00 00                                 ..
```

Extension blocks in general have the following format:

- Offset 0-1: Size of extension block
- Offset 2-3: Extension version
- Offset 4-7: Extension signature
- Offset 8+: Extension specific data

In the image above, we can see:

- Size = 0x48, or 72 decimal
- Version = 0x9
- Signature = 0x0400EFBE, or when converted BEEF0004 (start from the right and read left)
- After the signature the extension block specific data begins. BEEF0004 denotes a file entry extension block and as such we can recover creation and last access timestamps (FAT format), the version of Windows (Vista, 7, 8, etc.), long and short directory names, and MFT related information.

As SBE processes ShellBags, all of this binary data is interpreted and stored in different objects that make up the ShellBag, so rather than seeing a string of hexadecimal characters, SBE represents bags using common structures like Value, type, and so on (the fields, in general, as reflected in the grid view).

All available information about a bag is visible in details view, even when a bag contains data that cannot be abstracted into a grid view. For example, in some cases a bag may have 4 BEEF0004 blocks (and therefore 4 sets of timestamps, MFT info, etc.) and the details view allows you see all that data.

An example of the full details of a ShellBag representing a directory is shown below.

Details | Hex view

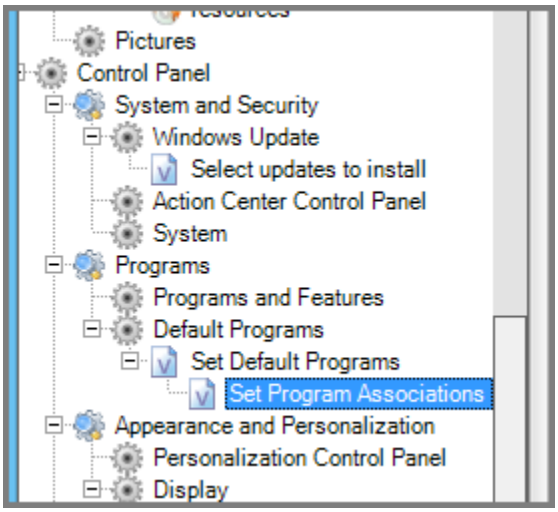
```
Short name: ADMINL~1
Modified: 10/3/2014 2:23:34 PM +00:00

Value: AdminLaptop
Shell Type: Directory

Bag Path: BagMRU\0\2\0\3\0, Slot #: 0, MRU Position: 1
Absolute Path: Desktop\My Computer\D:\Dropbox (Personal)\usrclass examples\HivesforSB\AdminLaptop

# Child Bags: 0

First explored: 11/17/2014 9:25:46 PM +00:00

Extension blocks found: 1
---------- Block 0 ----------
Signature: 0xbeef0004
Size: 72
Version: 9
Version Offset: 0x18

Identifier: 2E (Windows 8.1)

Created On: 10/3/2014 2:23:24 PM +00:00
Last Access: 10/5/2014 8:03:40 PM +00:00

Long Name: AdminLaptop

MFT Entry Number: 44
MFT Sequence Number: 59

File system hint: NTFS

---------------------------------------------------

Last Write Time: 11/17/2014 9:26:46 PM +00:00

Hex Value: 60-00-31-00-00-00-00-00-43-45-F1-72-10-00-41-44-4D-49-4E-4C-7E-31-00-00-48-00-09-00-04-00-EF-BE-43-45-EC-72-45-45-74-A0-
2E-00-00-00-2C-00-00-00-00-00-3B-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-88-A2-18-00-41-00-64-00-6D-00-69-00-6E-00-4C-00-61-00
-70-00-74-00-6F-00-70-00-00-00-18-00-00-00
```

Here we can see the short name, MAC dates and times, full bag path, the absolute path, and the first explored date.

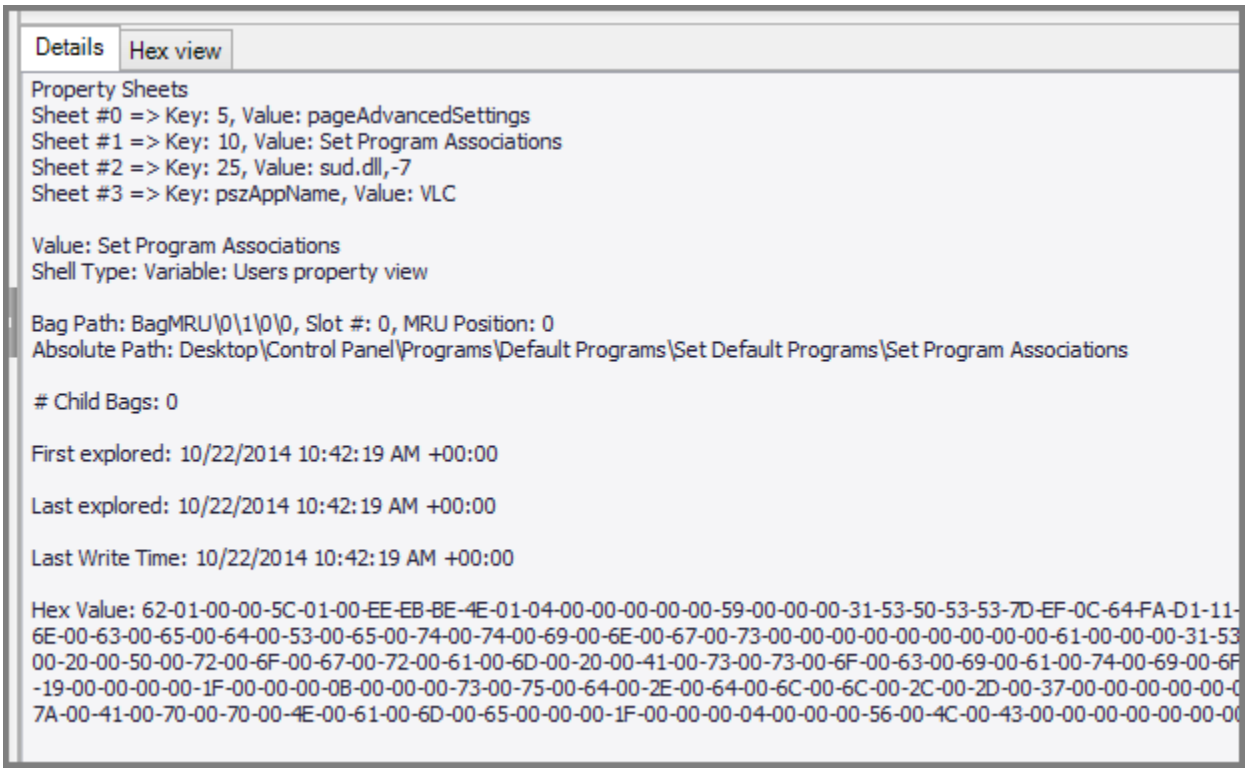
Note: The Last Write Time of the PARENT key (BagMRU\0\2\0\3\0) this ShellBag lives in is 11/17/2014 9:26:46 PM +00:00, but the First explored timestamp is 11/17/2014 9:25:46 PM +00:00. This is because First explored is determined by the last write date of the “bottom” key corresponding to BagMRU\0\2\0\3\0, Slot 0. The key for this bottom slot is not shown but the Last Write timestamp is collected and applied to the relevant bag.

Additionally we can see that the file system is NTFS. This can be determined since we have both an MFT entry and sequence number.

Some ShellBags contain embedded property sheets. These are another data structure commonly seen in various Windows artifacts and essentially are key/value pairs. In the example below, the bags related to the Control Panel have been expanded.



After selecting the "Set Program Associations" bag, its details view contains:

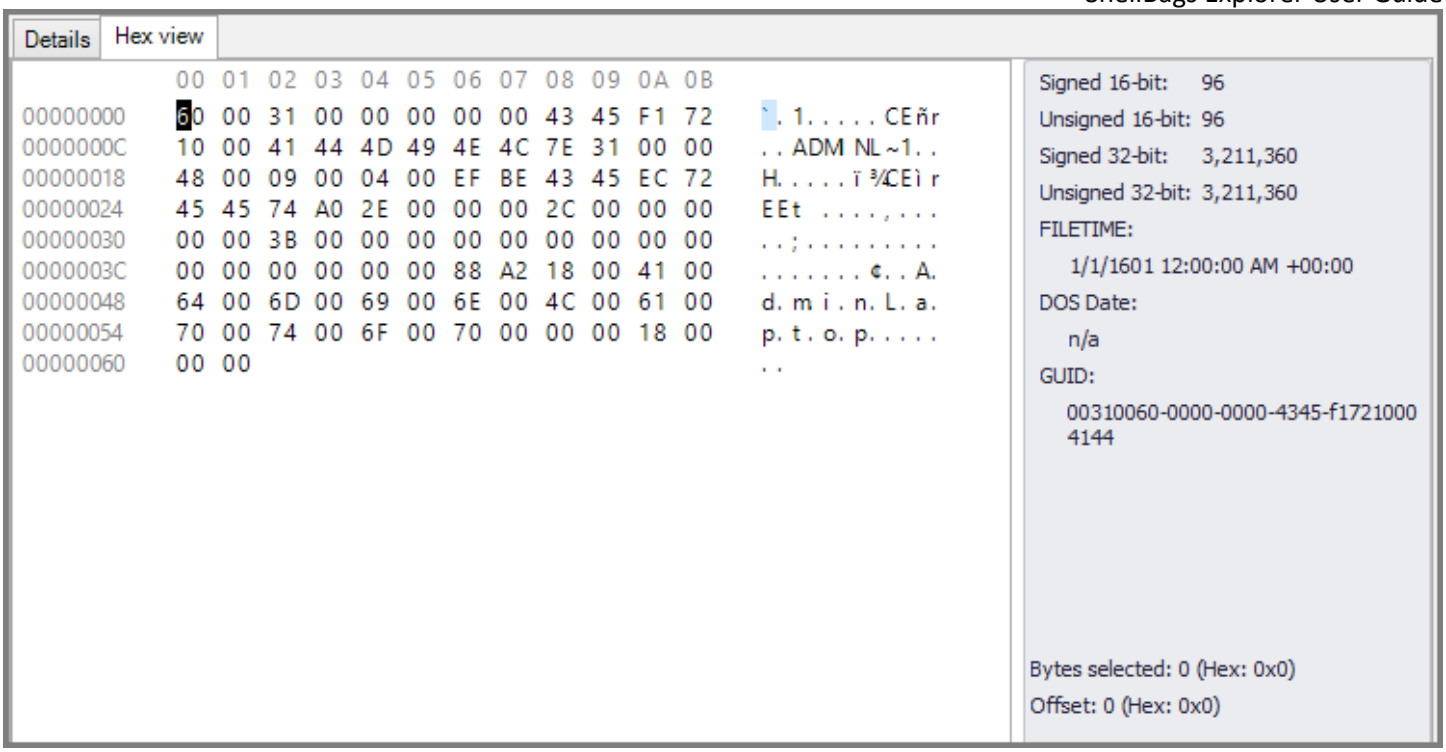


Here we can see the various property sheets and the data contained therein. Notice that now it is possible to tell not only WHEN program associations were changed, but WHAT program was used to take over those associations.

Finally, the bottom of the details view contains the raw hexadecimal data.

Hex view

The hex view contains the raw hexadecimal data of the selected Shellbag. It is displayed in traditional hex editor style in read-only mode.



To the right of the hex data is a rough interpretation of the data. A few strings can be seen in both ASCII and Unicode format. As the position of the cursor is changed by clicking on hex values, the data interpreter updates in real time.

Data interpreter

The data interpreter assists in decoding the contents of ShellBags when manual verification is desired. As the cursor is moved to different offsets, the interpreter updates in real time and shows different interpretations of the data FROM THE POSITION OF THE CURSOR IRRESPECTIVE OF ANY SELECTED BYTES.

Details | Hex view

```
          00 01 02 03 04 05 06 07 08 09 0A 0B
00000000  5A 00 31 00 00 00 00 00 98 44 24 69   Z. 1. . . . . . D$i
0000000C  10 00 4E 45 57 46 4F 4C 7E 31 00 00   . . NEWFOL~1. .
00000018  42 00 08 00 04 00 EF BE 98 44 24 69   B. . . . . ï ¾ D$i
00000024  98 44 24 69 2A 00 00 00 36 92 02 00   . D$i *. . . 6. . .
00000030  00 00 07 00 00 00 00 00 00 00 00 00   . . . . . . . . . . . .
0000003C  00 00 00 00 00 00 4E 00 65 00 77 00   . . . . . . N. e. w.
00000048  20 00 66 00 6F 00 6C 00 64 00 65 00    . f. o. l. d. e.
00000054  72 00 00 00 18 00 00 00               r. . . . . . .
```

Signed 16-bit: 17,560
Unsigned 16-bit: 17,560
Signed 32-bit: 1,763,984,536
Unsigned 32-bit: 1,763,984,536
FILETIME:
n/a
DOS Date:
4/24/2014 1:09:08 PM +00:00
GUID:
69244498-0010-454e-5746-4f4c7e310000

Bytes selected: 0
Offset: 8 (Hex: 0x8)

(Double click here to copy values)

In this example the cursor is at offset 0x8 and we see the DOS date is showing 4/24/2014 1:09:08 PM +00:00 which corresponds to the Modified timestamp.

In the next example we can see the data interpreter has found a GUID for the Control Panel at offset 0x4.

Details | Hex view

```
          00 01 02 03 04 05 06 07 08 09 0A 0B
00000000  14 00 1F 70 68 06 EE 26 0A A0 D7 44   . . . ph. î &   ×D
0000000C  93 71 BE B0 64 C9 86 83 00 00         . q³/ª dÉ. . . .
```

Signed 16-bit: 1,640
Unsigned 16-bit: 1,640
Signed 32-bit: 653,133,416
Unsigned 32-bit: 653,133,416
FILETIME:
n/a
DOS Date:
3/8/1983 4:55:28 AM +00:00
GUID:
26ee0668-a00a-44d7-9371-beb064c98683 (Maps to: Control Panel)

Bytes selected: 0
Offset: 4 (Hex: 0x4)

(Double click here to copy values)

By using the data interpreter to explore ShellBags you can begin to see common patterns including timestamps, extension blocks, property sheets, and so on.

NOTE: The data interpreter will show certain fields in **bold** based on certain conditions:

- A GUID is found that maps to a known location
- A timestamp where the calculated datetime > Now - 10 years AND calculated datetime < Now + 5 years.

These provide visual cues that a valid timestamp or GUID may have been found.

Finally, the data from the data interpreter and its corresponding ShellBag details can be copied to the clipboard by double clicking the designated area at the bottom of the data interpreter. An example is shown below of what this data may look like once copied into a text editor.

```
    0........10........20........30........40........50........60........70......
 0  Offset: Offset: 4 (Hex: 0x4)
 1
 2
 3  Signed 16-bit: 1,640
 4  Unsigned 16-bit: 1,640
 5  Signed 32-bit: 653,133,416
 6  Unsigned 32-bit: 653,133,416
 7  FILETIME: n/a
 8  DOS Date: 3/8/1983 4:55:28 AM +00:00
 9  GUID: 26ee0668-a00a-44d7-9371-beb064c98683 (Maps to: Control Panel)
10
11
12  Value: Control Panel
13  Shell Type: Root folder: GUID
14
15  Bag Path: BagMRU, Slot #: 2, MRU Position: 11
16  Absolute Path: Desktop\Control Panel
17
18  # Child Bags: 3
19
20  Hex Value: 14-00-1F-70-68-06-EE-26-0A-A0-D7-44-93-71-BE-B0-64-C9-86-83-00-00
21
22
23
```

SBECmd.exe

SBECmd.exe is a command line version of ShellBags Explorer that will automatically process hive files and export the results to TSV. The command line options are shown below.

```
SBECmd 0.5.0.0
Copyright (C) 2014 Eric R. Zimmerman (saericzimmerman@gmail.com)


Usage: SBECmd -d <directory>

  -d             Required. Directory file to look for registry hives

  --dedupe       (Default: False) When true, SBECmd processes all hives and
                 removes duplicate shellbag items. Deduplication is based on
                 BagPath, Slot, MRUPosition, Value, CreatedOn, ModifiedOn,
                 AccessedOn, MFTEntry, MFTSequenceNumber, FirstExplored, and
                 LastExplored

  --timezone     (Default: GMT Standard Time) The timezone to use when
                 displaying dates and times (Default = GMT Standard Time).
                 Enclose in quotes. Use --timezone="" to see a list of all
                 available timezones

  --help         Display this help screen.


Using -d, SBECmd will process each file in <directory> and create one TSV
report per file found in <directory>\Out.

Using -d and --dedupe, SBECmd will process each file in <directory> and create
one report containing the information from ALL hives with duplicated shellbags
removed.
```

Processing multiple, unrelated hives

After exporting one or more hives to a directory, execute SBECmd.exe as follows:

SBECmd.exe -d <directory>

A new directory underneath <directory> will be created called 'Out' which will contain the TSV reports generated by SBECmd.exe. An example run is shown below.

```
C:\temp>SBECmd.exe -d c:\Users\eric\Desktop\aa

SBECmd.exe version 0.5.0.0

Directory to process: c:\Users\eric\Desktop\aa
Export format: TSV
Deduplication: False
Timezone: GMT Standard Time

Processing 'c:\Users\eric\Desktop\aa\UsrClass 1.dat'
Parse time: 0.33 seconds

        Total Shell Bags found: 65

        Totals by bag type

        Directory: 56
        Drive letter: 4
        Root folder: GUID: 3
        GUID: Control panel: 1
        Control Panel Category: 1

Parse time: 0.33 seconds

        Total Shell Bags found: 65

        Totals by bag type

        Directory: 56
        Drive letter: 4
        Root folder: GUID: 3
        GUID: Control panel: 1
        Control Panel Category: 1

Finished processing 'c:\Users\eric\Desktop\aa\UsrClass 1.dat'

Exported to: 'c:\Users\eric\Desktop\aa\Out\UsrClass 1.dat.tsv'

----------------------------------------------------------------------
Processing 'c:\Users\eric\Desktop\aa\UsrClass 2.dat'
Parse time: 0.60 seconds

        Total Shell Bags found: 136

        Totals by bag type

        Directory: 121
        Drive letter: 6
        Root folder: GUID: 6
        GUID: Control panel: 1
        Control Panel Category: 1
        Users Files Folder: 1

Parse time: 0.60 seconds

        Total Shell Bags found: 136

        Totals by bag type

        Directory: 121
        Drive letter: 6
        Root folder: GUID: 6
        GUID: Control panel: 1
        Control Panel Category: 1
        Users Files Folder: 1

Finished processing 'c:\Users\eric\Desktop\aa\UsrClass 2.dat'

Exported to: 'c:\Users\eric\Desktop\aa\Out\UsrClass 2.dat.tsv'

----------------------------------------------------------------------

Processing complete!
Processed 2 files in 0.93 seconds!
Total Shell Bags found: 201
```

If SBECmd.exe encounters any errors processing hives, they will be reported to the console and to '!SBECmd_Messages.txt' in <directory>\Out

Processing multiple, related hives

In certain cases you will have multiple registry hives from the same user. These hives may be found in volume shadow copies, carving, etc.

SBECmd can also deduplicate ShellBags across multiple registry hives via the --dedupe switch. This is useful when processing many hives from a given user and eliminating duplicate shellbag entries. Uniqueness is determined from the following data in a ShellBag: BagPath, Slot, MRUPosition, Value, CreatedOn, ModifiedOn, AccessedOn, MFTEntry, MFTSequenceNumber, FirstExplored, and LastExplored. These values are combined and then SHA-1 hashed. Only one copy of a ShellBag with a given SHA-1 hash is reported and the rest are discarded.

After exporting one or more hives to a directory, execute SBECmd.exe as follows:

SBECmd.exe -d <directory> --dedupe

A new directory underneath <directory> will be created called 'Out' which will contain the TSV report named 'Deduplicated.tsv'. An example run is shown below.

```
C:\temp>SBECmd.exe -d c:\Users\eric\Desktop\aa --dedupe

SBECmd.exe version 0.5.0.0

Directory to process: c:\Users\eric\Desktop\aa
Export format: TSV
Deduplication: True
Timezone: GMT Standard Time

Processing 'c:\Users\eric\Desktop\aa\UsrClass 1.dat'
--------------------------------------------------------------------
Processing 'c:\Users\eric\Desktop\aa\UsrClass 2.dat'
--------------------------------------------------------------------
Unique Shell Bags found: 162 (19.40 % duplicates)
Results saved to 'c:\Users\eric\Desktop\aa\Out\Deduplicated.tsv'

Processing complete!
Processed 2 files in 0.93 seconds!
Total Shell Bags found: 201
```

Time zone support

Like its GUI counterpart, SBECmd can adjust all dates and times to a user selected time zone. The default time zone is UTC, but a different time zone can be selected via the –timezone switch. Enclose the timezone in quotes. An example is shown below.

```
C:\temp>SBECmd.exe -d c:\Users\eric\Desktop\aa --dedupe --timezone="Mountain standard time"

SBECmd.exe version 0.5.0.0

Directory to process: c:\Users\eric\Desktop\aa
Export format: TSV
Deduplication: True
Timezone: Mountain standard time

Processing 'c:\Users\eric\Desktop\aa\UsrClass 1.dat'
--------------------------------------------------------------------
Processing 'c:\Users\eric\Desktop\aa\UsrClass 2.dat'
--------------------------------------------------------------------
Unique Shell Bags found: 162 (19.40 % duplicates)
Results saved to 'c:\Users\eric\Desktop\aa\Out\Deduplicated.tsv'

Processing complete!
Processed 2 files in 0.93 seconds!
Total Shell Bags found: 201
```

The TSV file(s) will now reflect the selected time zone, including any adjustments for daylight savings time.

kCount	O FirstExplored	P LastExplored	Misce
0			
0	11/7/2014 10:29:55 AM -07:00		
0		11/13/2014 8:42:44 AM -07:00	
0			
0		11/8/2014 9:54:49 AM -07:00	
0			
0			
1			NTFS
1	11/7/2014 10:03:32 AM -07:00		NTFS
1			exFAT
1	11/7/2014 3:25:07 PM -07:00		FAT fi
1	11/7/2014 3:25:08 PM -07:00	11/7/2014 3:25:08 PM -07:00	FAT fi
1	11/7/2014 10:02:59 AM -07:00	11/7/2014 10:02:59 AM -07:00	NTFS
1	11/7/2014 3:22:48 PM -07:00		exFAT
1	11/7/2014 3:23:51 PM -07:00	11/7/2014 3:23:51 PM -07:00	exFAT
1		11/8/2014 9:28:00 AM -07:00	NTFS
1			NTFS
1			NTFS
1		11/7/2014 12:30:46 PM -07:00	NTFS
1			NTFS
1			NTFS
1	11/7/2014 12:30:33 PM -07:00		NTFS
1	11/7/2014 12:30:38 PM -07:00		NTFS
1	11/7/2014 12:30:43 PM -07:00		NTFS
1			NTFS
1			
1		11/8/2014 3:19:56 PM -07:00	NTFS
1	11/7/2014 10:33:31 AM -07:00	11/7/2014 3:17:04 PM -07:00	NTFS
1	11/7/2014 11:03:09 AM -07:00		NTFS
1	11/7/2014 11:03:12 AM -07:00		NTFS
1	11/7/2014 12:29:56 PM -07:00	11/7/2014 12:29:56 PM -07:00	

General usage tips and tricks

(To be completed later)

Version changes

Version 0.7.0.0

NEW: Updated controls
NEW: Open hives via right click | Open or as an external application from programs like X-Ways, etc.
NEW: Detect if .net 4.6 is installed
NEW: A bunch of new GUIDs
NEW: Support additional type IDs in XP hives
NEW: Added support for Beef0005 extension blocks (which contain multiple shell items and extension blocks themselves)
NEW: Handle lots of other fringe things
NEW: CTRL-C copies selected bytes to clipboard in Hex view
CHANGE: Show version # in title bar
CHANGE: Add fallback search for BagMRU key if standard ones aren't found
CHANGE: Save/restore width of tree
FIX: Handle more types of zip file contents in XP hives
FIX: Correct issue showing Last Access time from zip file contents.
FIX: Handle Unicode better in CDBURN shell bags
FIX: Handle Unicode better in directory shell bags

Version 0.6.1.0

NEW: Added support for Windows XP ShellBags in NTUSER.DAT hives
CHANGE: Updated controls

Version 0.6.0.0

NEW: Added NodeSlot info. This column is hidden by default in the grid, but will always be shown in Details view
CHANGE: Updated Registry parser code (~150% faster and significant memory reduction)
FIX: Correct FAT vs exFAT file system hint in BEEF0004 blocks
FIX: Correctly detect end of ShellBag in C3 bags

Version 0.5.4.0

CHANGE: Replace custom logging code with NLog
FIX: A few cosmetic fixes to ShellBag details output
FIX: Registry parser bug fixes on some fringe cases

Version 0.5.3.0

(Internal testing version)

Version 0.5.2.0

NEW: Detect deleted ShellBags. When deleted ShellBags are found they will have a red X after the name of the bag.
NEW: Added GUID for Network Neighborhood

CHANGE: Expanded coverage for MTP type 2 devices (a lot of new detail available in Details view)
CHANGE: Expanded coverage for MTP type 1 devices (a lot of new detail available in Details view)

FIX: Cosmetic clean up in SBECmd output (showing results twice)

Version 0.5.1.0

NEW: Added GUID for Hyper-V Remote file browsing
NEW: Added support for type 0x02 ShellBags, seen when browsing Hyper-V servers
NEW: General support for browsing Hyper-V resources including server, drive letter, and paths (including MAC timestamps for folders)

CHANGE: Replace woanware registry parsing code with my own

Version 0.5.0.5

NEW: Change alternate rows to a different color for visual separation of data

CHANGE: Save window size after it is resized vs only on exit
CHANGE: You can now select multiple cells vs rows only in the grid. This lets you copy just values or timestamps vs getting all the data

FIX: Prevent crash when right clicking on a group by row

Version 0.5.0.4

NEW: Build for AnyCPU vs forcing x86.
NEW: Add icon for "History folder" ShellBag type
NEW: Allow hives to be dropped on Hex view
NEW: Added millisecond precision to timestamps that have that level of resolution. These timestamps are visible in the details pane
NEW: Added support for http URLs in variable blocks.
NEW: Added ability to double click on Data Interpreter to copy values on Interpreter plus ShellBag details to Clipboard
NEW: FILETIME and DOS Date fields in data interpreter will be bolded if the calculated date > Now - 10 years AND calculated date < Now + 5 years. This provides a visual cue a valid timestamp may have been found.
NEW: GUID field in the data interpreter will be bolded if the calculated GUID maps to a known value
NEW: Count all ShellBag registry values seen and compare to the number of ShellBags processed. If seen != processed, show warning on summary screen as data is missing
NEW: Added Placeholder extension block which is used to "pad" main ShellBags when they contain additional ShellBag items (Bags containing bags). Placeholder bags are not printed to the details pane
NEW: Added detection of Beef0010 blocks in 0x71 ShellBags.
NEW: Added detection of Beef0010 blocks in 0x2f ShellBags.
NEW: Added ability to double click on a row in the Messages window which will select the related node in the tree on the main window
NEW: Add GUID for OneDrive on Windows 10
NEW: Detect drive letters in 0x1F ShellBags.

CHANGE: Detect optional Beef0004 extension block in CDBurn ShellBags
CHANGE: Look for common signatures in several bags and act accordingly

FIX: Detect several fringe cases and set value to "!!! Unable to determine value !!!". These need more research for full support
FIX: Remove early return in 0x4d bag which resulted in the ShellBag not displaying properly

Version 0.5.0.3

NEW: Add new ShellBag type, ShellBagParseError, that is used as a placeholder when ShellBag contents aren't parsed properly.

Version 0.5.0.2

NEW: Added support for 0x61 bags (FTP). Connection date and username (when saved) are reported
NEW: Added support for Variable: FTP URI ShellBags. These show directories browsed on an FTP server and in many cases contains the timestamp on the FTP folder

CHANGE: More support for contents in BEEF000e extension blocks
CHANGE: Improve message when BagMRU key isn't found in a hive.
CHANGE: Updated drive letter icon to a hard drive vs green folder
CHANGE: Expand and collapse nodes now affects the selected node vs all nodes.

FIX: Some GUI fixes (when using expand/collapse nodes remove recursive icon)

Version 0.5.0.1

FIX: Remove recursive icon when right clicking an entry in the grid
FIX: Minor manual updates

Version 0.5.0.0

NEW: Completely rewritten manual
NEW: Added Time zone setting to Tools menu. Change the time zone BEFORE processing a hive to have all dates automatically adjusted for the selected timezone
NEW: Added --timezone command line argument to SBECmd.exe. Use --timezone="" for a list of valid time zones
NEW: Added --dedupe option to SBECmd. This will process all hives in a directory and remove any duplicate bags. See --help for details

FIX: Adjust Beef0025 version offset to always pull the right offset.

Version 0.4.3.0

CHANGE: Switch from CSV extension to TSV for easier processing in Excel without having to import manually

FIX: Minor fixes to command line version if source directory is missing

Version 0.4.2.0

FIX: Correct file system hint in details pane misreporting file system. The value in Misc column was correct but in some cases the details pane misreported the file system type

Version 0.4.1.0

NEW: A shiny manual is now included
NEW: CSV output has been added. You can optionally include the raw hex from bags as well
NEW: Added SBECmd.exe which is the command line version of SBE.
NEW: Added FirstExplored column to grid and details section (when available)
NEW: Added LastExplored column to grid and details section (when available)
NEW: More GUIDs (Windows 10, RealPlayer Cloud, etc)
NEW: Added support for ShellBags with TypeID 0x01 special case
NEW: You can now drag and drop an offline hive on the grid, tree, or details pane to load
NEW: Added note to Miscellaneous column denoting what file system a bag came from based on MFT information (Thanks to David Cowen for idea and testing)

Version 0.3.9.0

NEW: Added icon for CDBurn bags
NEW: Added CTRL-A and CTRL-C functionality to main grid. This lets you quickly select all rows and copy selected rows to clipboard
NEW: SBE now remembers its size and will restore the last size on startup

CHANGE: Add detection of certain special signatures earlier in the process as these types of bags can occur in different places (CDBurn, etc)
CHANGE: Rename "Variable: CDBurn" to "CDBurn" since they can be seen in different places

Version 0.3.8.0

NEW: Added operating system information for beef0004 blocks based on Identifier property. 0x14 => Windows XP, 2003 | 0x26 => Windows Vista | 0x2a => Windows 2008, 7, 8.0 | 0x2e => Windows 8.1
NEW: Report unmapped GUIDs to messages window
NEW: Added support for 0x35 identifiers, which contain Unicode strings vs ascii
NEW: Added support for ShellBag with type ID 0x64 and 0x65. These refer to browser history folders
NEW: Even MORE GUIDS!!!!!1

CHANGE: Improvements to message form details pane
CHANGE: Hide InternalID and HexView from grid field chooser

FIX: Deal with improperly terminated Unicode strings so a ? isn't displayed as the last character
FIX: Handle embedded property stores in 0x71 (Control panel bags) vs reporting unknown GUID
FIX: Lots of fringe case cleanup

Version 0.3.7.0

NEW: Report unknown type IDs to Messages window so they can be reported
NEW: Added support for Beef0010 extension block. This block contains property store data. Vector data in one of the sheets looks to contain additional property stores with additional data. More work to be done here
NEW: Added support for Beef0021 extension block. This block has been seen to contain the URL where files have been downloaded from
NEW: Added support for Beef0016 extension block. This looks to contain what was searched for (ie *.inf)

CHANGE: Do not force 'Last write time' column visible when loading an offline hive
CHANGE: Added more GUIDs

FIX: Don't crash when grouping and a group is selected

Version 0.3.6.0

NEW: Pull extension blocks out of property sheets in 0x00 and 0x1F ShellBag types. These appear to be the results of searches. More research is needed to confirm

CHANGE: Clean up some messages

Version 0.3.5.0

NEW: Added 'Absolute path' to grid which contains the full path from the Desktop to the bag item
NEW: Added a few dozen more GUIDs
NEW: Alternate row colors in Messages window
NEW: Added exporting of messages to Excel via button on Messages window

NEW: Added 'Clear messages' button to Messages window

CHANGE: Made column headers look nicer by adding spaces (it's the little things sometimes)
CHANGE: Tweak Messages window to allow for message type, message, and details view to unclutter the display
CHANGE: Recoded Root folder shell item ShellBag type (0x1f) to find and add multiple extension blocks
CHANGE: Expand hex view to fill more space and move data interpreter to far right
CHANGE: Show wait cursor while doing things vs arrow
CHANGE: Add detection of bags related to zip file contents early vs inside each known bag type. This prevents a lot of future "unknown" types going forward

Appendix A – Contributors

The following people have contributed in one way or another during the development and refinement of SBE

- SA/FE Devon P. Ackerman devon.ackerman@ic.fbi.gov, sadevonackerman@gmail.com
- David Cowen @HECFBlog
- Harlan Carvey @keydet89
- Dan Pullega @4n6k
- Mark Woan @woanware

Appendix B – Additional resources

- http://msdn.microsoft.com/en-us/library/aa965725%28v=vs.85%29.aspx
- https://code.google.com/p/libfwsi/wiki/ShellFolderIdentifiers
- https://docs.google.com/file/d/0B-VYGsDJPtZlVDNJQ3pWX0M1b1k/edit
- http://www.4n6k.com/2013/12/shellbags-forensics-addressing.html
- http://www.williballenthin.com/forensics/shellbags/index.html