



ShellBags Explorer

Eric R. Zimmerman
saericzimmerman@gmail.com
501-313-3778

Revision history

11/21/2014 Rev. 1 – Initial release

01/20/2017 Rev. 2 – Version 0.8.0.0

Table of Contents

Requirements.....	5
What are ShellBags?	5
ShellBags location in the Registry	5
Using RegEdit to view ShellBag data.....	6
ShellBag binary contents.....	6
MRUListEx	8
Subkeys and their relation to values.....	8
LastWrite times and their value in ShellBags analysis	10
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	15
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	20
Workflow overview.....	20
Messages window	21
Tree view	23
Grid view	27

Details view	36
Hex view	40
SBECmd.exe	43
Processing multiple, unrelated hives	43
Processing multiple, related hives	44
Time zone support	47
General usage tips and tricks	48
Version changes	49
Version 0.8.0.0	49
Version 0.7.0.0	49
Version 0.6.1.0	49
Version 0.6.0.0	49
Version 0.5.4.0	50
Version 0.5.3.0	50
Version 0.5.2.0	50
Version 0.5.1.0	50
Version 0.5.0.5	50
Version 0.5.0.4	50
Version 0.5.0.3	51
Version 0.5.0.2	51
Version 0.5.0.1	51
Version 0.5.0.0	51
Version 0.4.3.0	51
Version 0.4.2.0	52
Version 0.4.1.0	52
Version 0.3.9.0	52
Version 0.3.8.0	52
Version 0.3.7.0	52
Version 0.3.6.0	53
Version 0.3.5.0	53
Appendix A – Contributors.....	53
Appendix B – Additional resources.....	53

Requirements

ShellBags Explorer requires Microsoft .net framework version 4.6 full runtime or greater to be installed. It is available at <https://www.microsoft.com/en-us/download/details.aspx?id=53344>.

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 in the following Registry keys and subkeys:

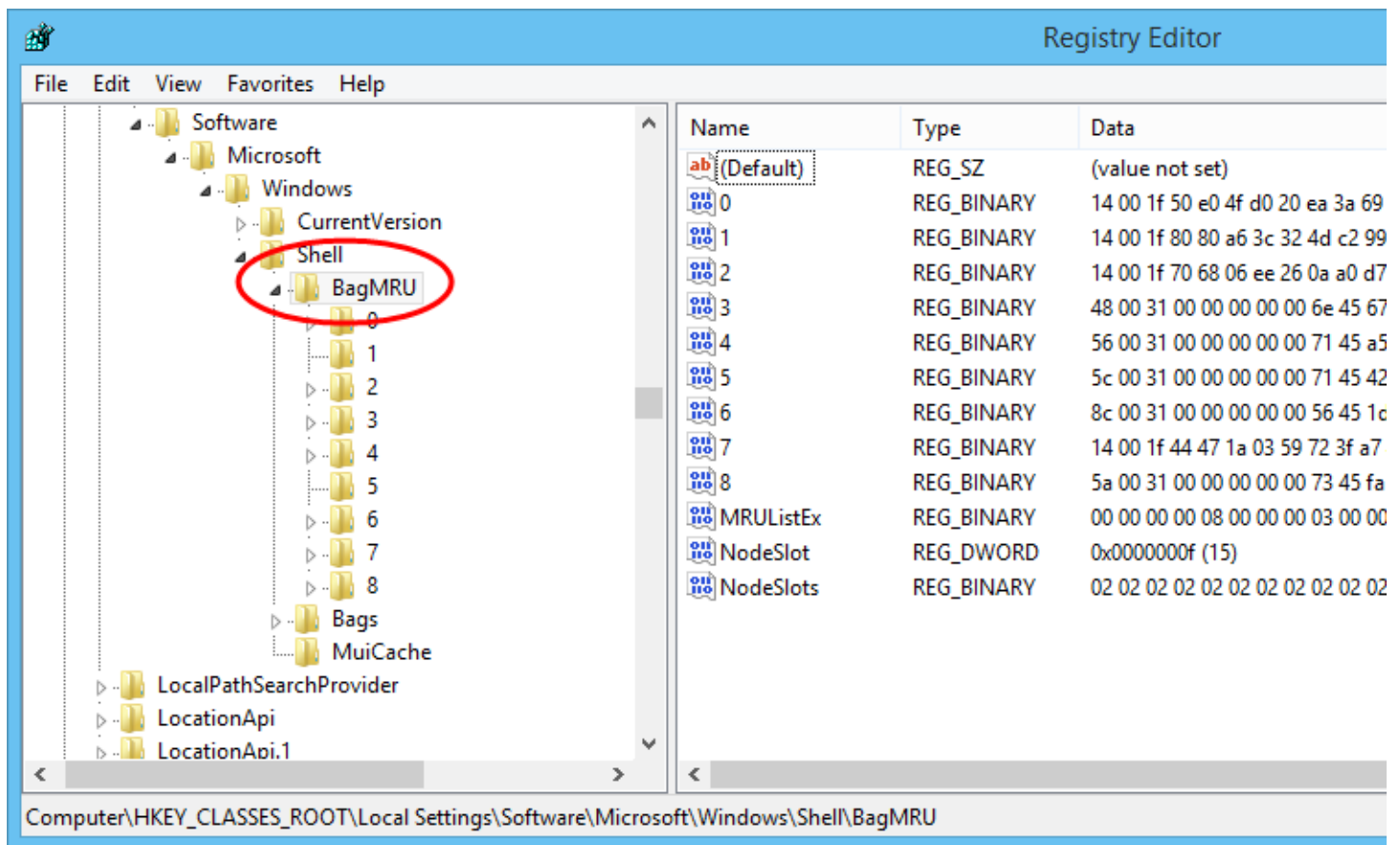
- HKEY_CURRENT_USER\Software\Microsoft\Windows\Shell\Bags
- HKEY_CURRENT_USER\Software\Microsoft\Windows\Shell\BagMRU (ntuser.dat)
- HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\Bags
- HKEY_CURRENT_USER\Software\Microsoft\Windows\ShellNoRoam\BagMRU (ntuser.dat)
- HKEY_CURRENT_USER\Software\Classes\Local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

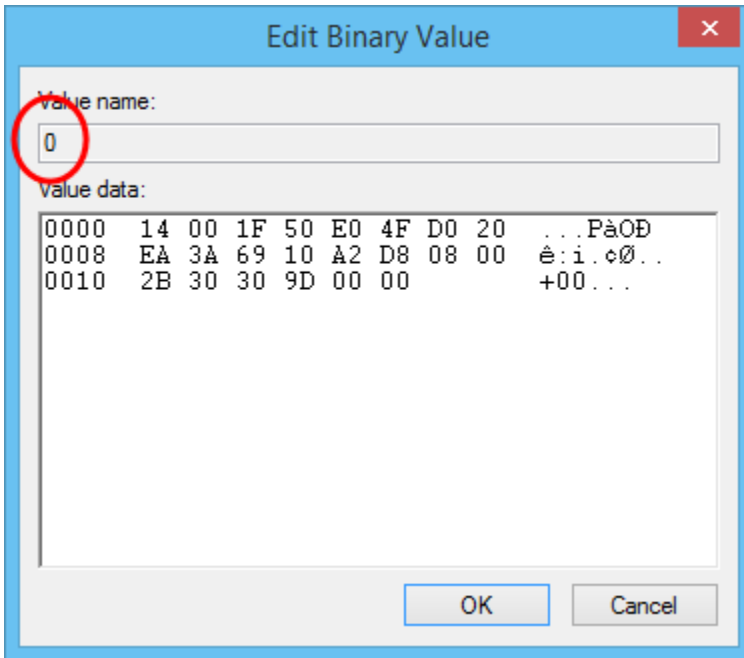


On the left, Registry keys are displayed. After selecting a key, the values for that key are displayed in the right pane.

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.

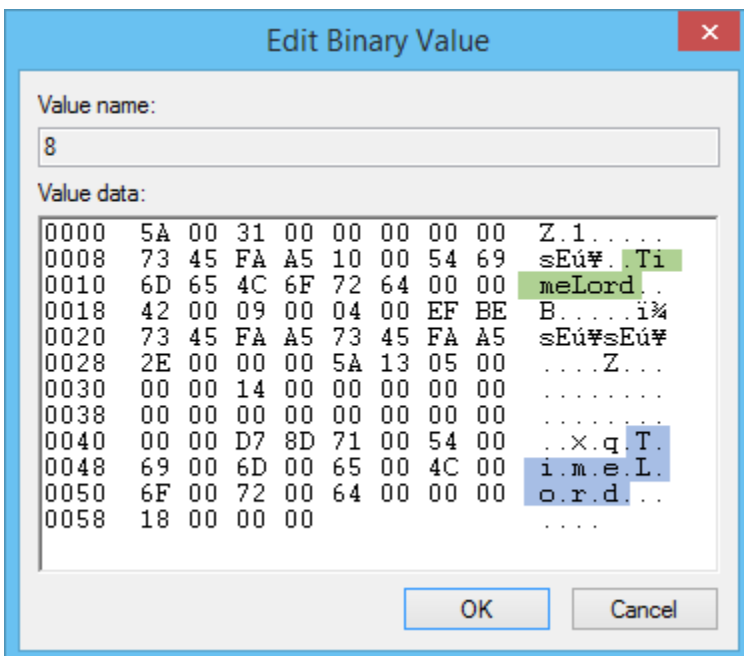
ShellBag binary contents

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.

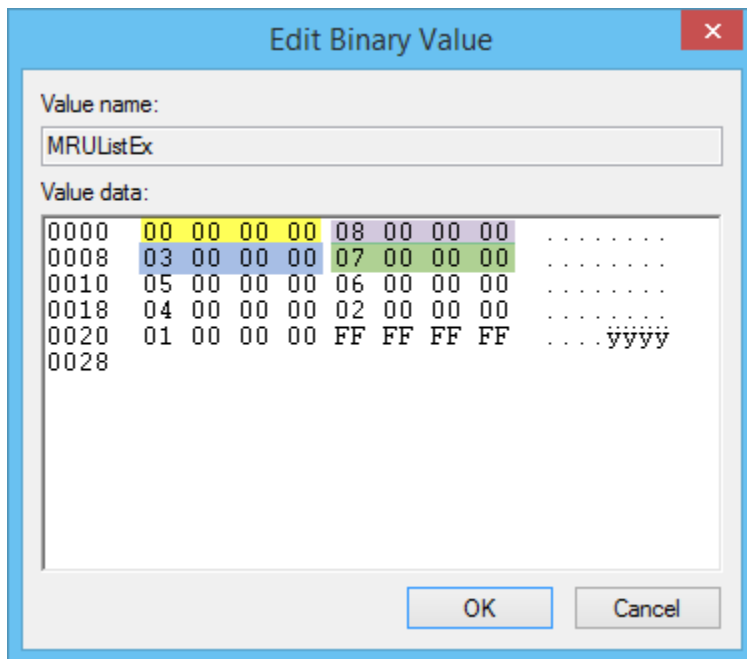


While not obvious, there are also several dates and times embedded in the binary data as well as MFT entry and sequence numbers. It is also possible to determine the file system this directory existed on based on the contents of the ShellBag. In the example above, the directory existed on an NTFS file 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.



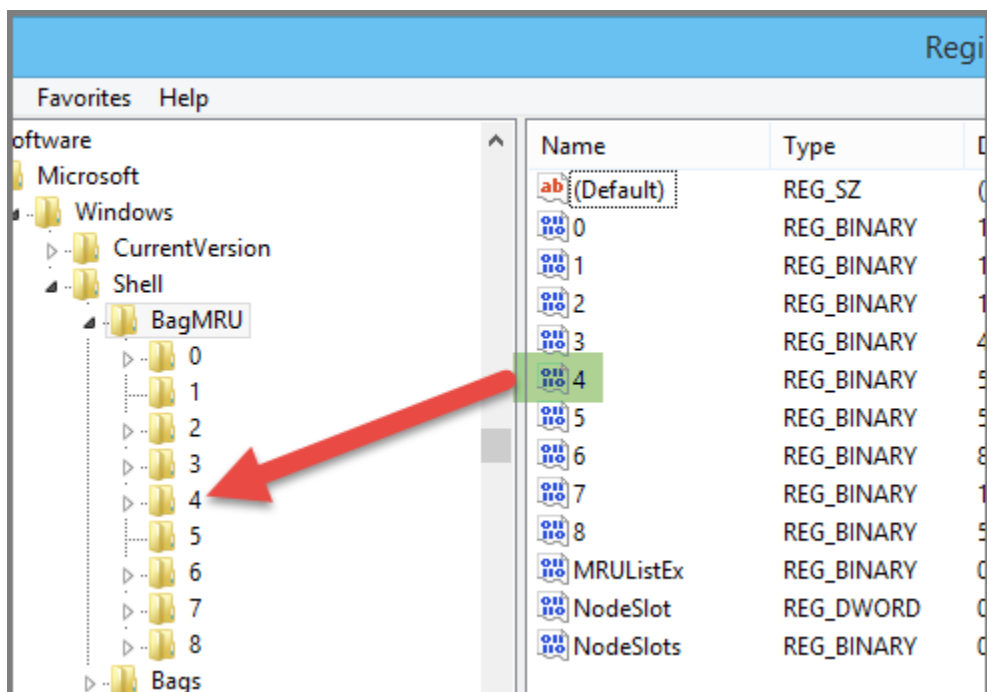
Each entry in the MRU list is 4 bytes long (little endian). In the image above, the first 4 MRU positions have been highlighted with different colors.

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.

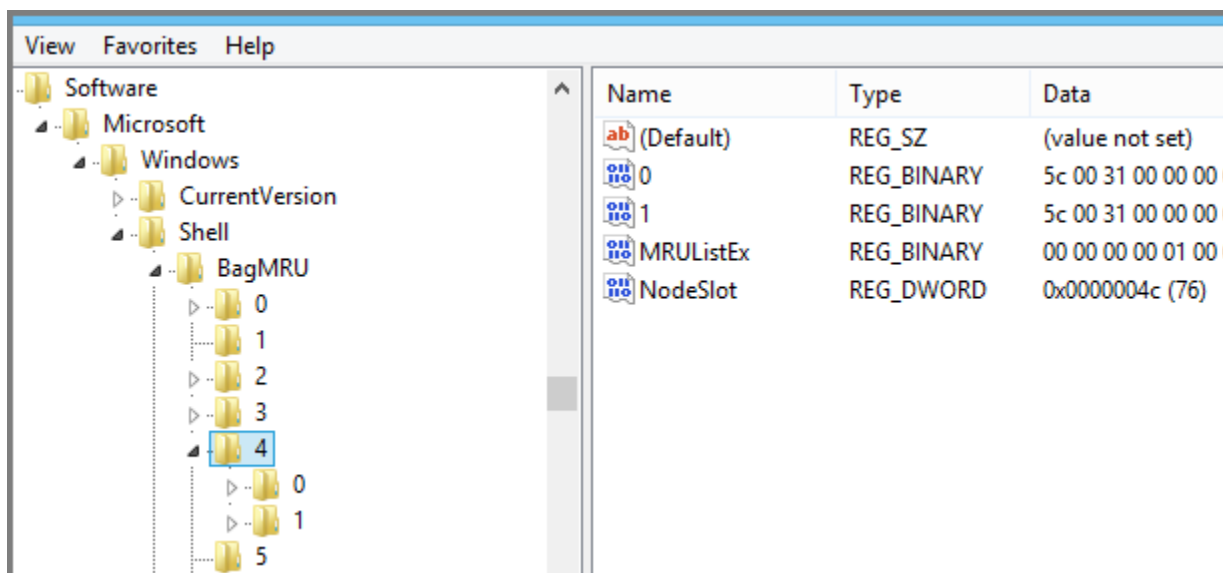
Subkeys and their relation to values

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.

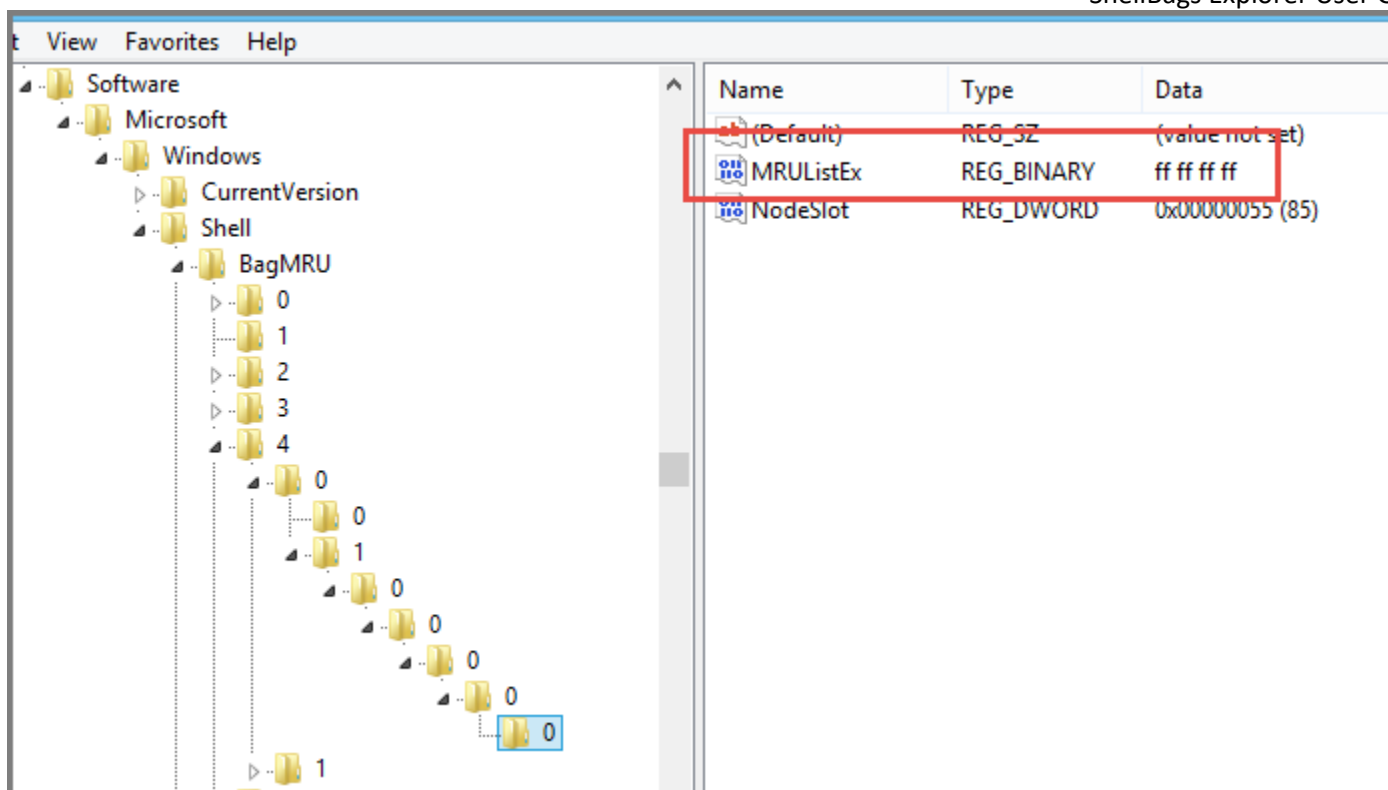
Subkeys with the same name as the value will contain child objects for that value. Recall each value represents a ShellBag. A ShellBag can be the Desktop item, a Control Panel Category, a Control Panel item, a drive letter, or a directory (there are other possibilities as well).



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 Registry Explorer can. When using Registry Explorer to look at a similar key as shown above, we can see the key was last written on 2015-07-11 14:40:50 +00:00.

Registry Explorer v0.8.1.0

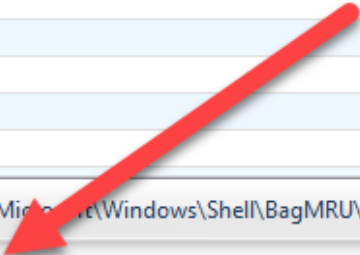
File Tools Options Bookmarks (2/0) View Help

Registry hives (1) Available bookmarks (4/0)

Key name	# values	Last write timestamp
{AFA7FF39-1DDF-4F70-A2D5-23FCFFF02E5F}	1	2014-11-29 05:21:25 +0...
{E91579C0-4EA9-4A2A-A9B2-04BEF1D6DC29}	1	2015-05-26 02:44:22 +0...
Shell	0	2014-11-01 04:35:39 +0...
BagMRU	12	2015-08-03 02:09:13 +0...
0	10	2015-08-03 02:04:20 +0...
0	5	2015-07-11 14:40:53 +0...
0	2	2014-11-01 17:50:04 +0...
1	6	2015-07-11 14:40:50 +0...
2	2	2015-07-11 14:40:53 +0...
1	2	2014-11-01 04:03:21 +0...
2	3	2015-04-25 22:09:33 +0...
3	14	2015-07-28 14:00:30 +0...
4	3	2014-11-01 17:50:16 +0...
5	4	2015-07-18 04:44:41 +0...

Key: Local Settings\Software\Microsoft\Windows\Shell\BagMRU\0\0\1

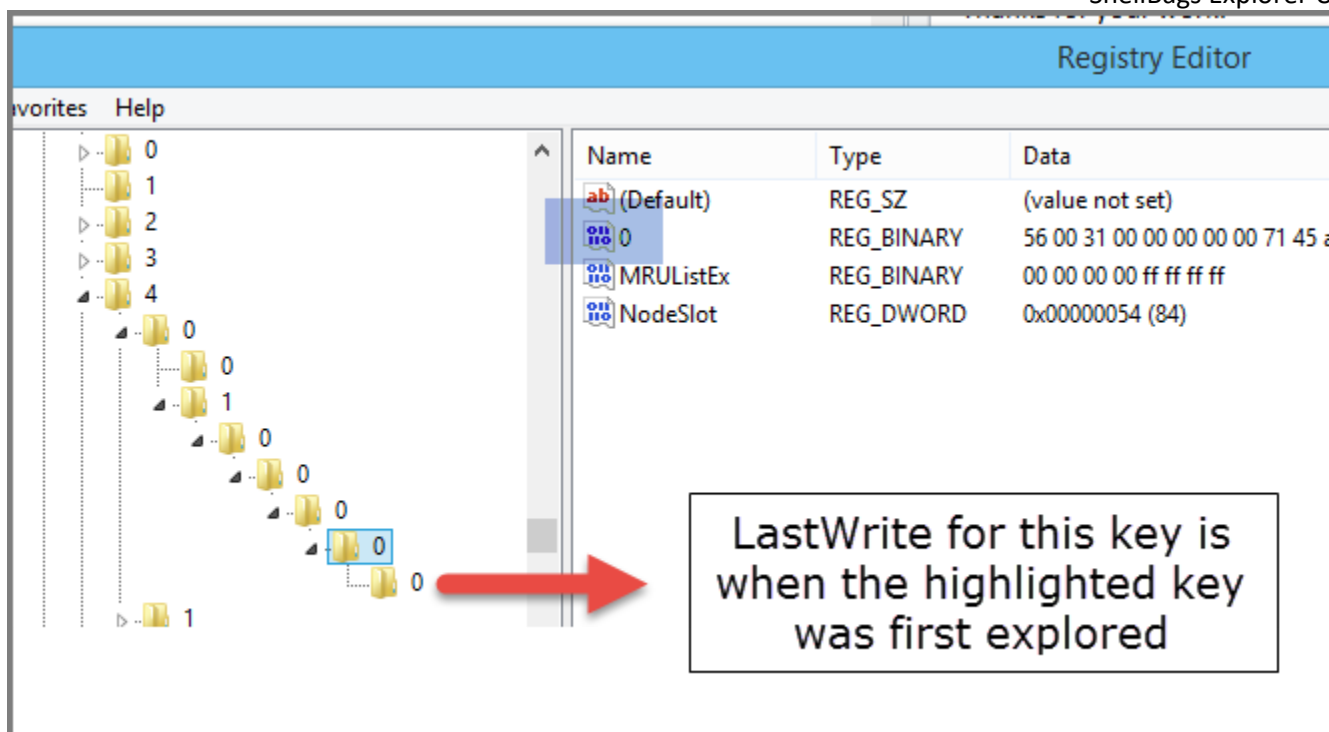
Last write: 2015-07-11 14:40:50 +00:00 6 of 6 values shown (100.00 %) Copied last write time to clipboard



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

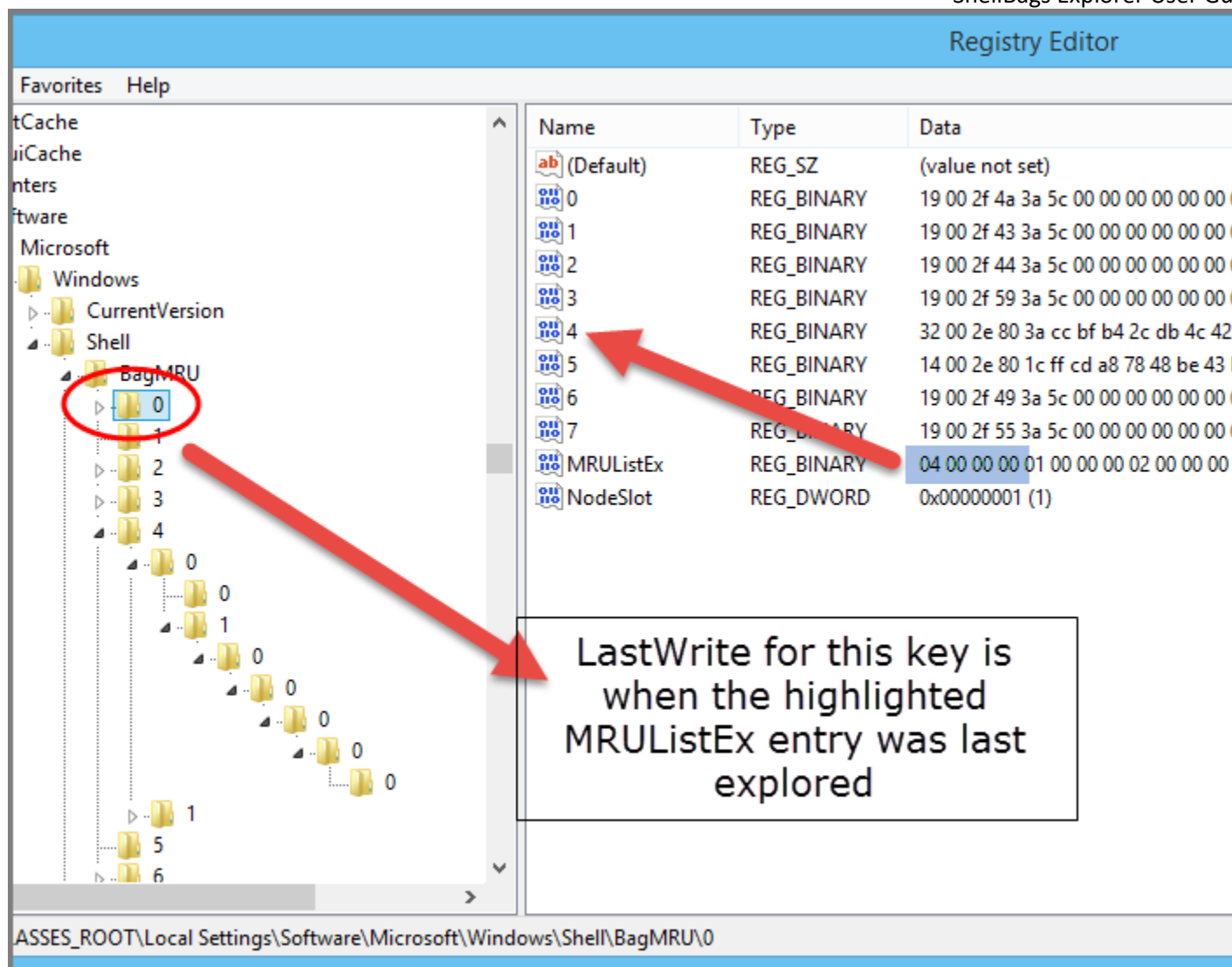
Keeping in mind how LastWrite timestamps work, we can now determine when a given ShellBag was accessed.

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.) like 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.
- When using the command line tool, the 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 devices)
- 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 <https://goo.gl/6dl4FS>. The specification outlines all known shellbag types and layouts and is continually updated by Metz as new information is discovered.
- Certain shellbags 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 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

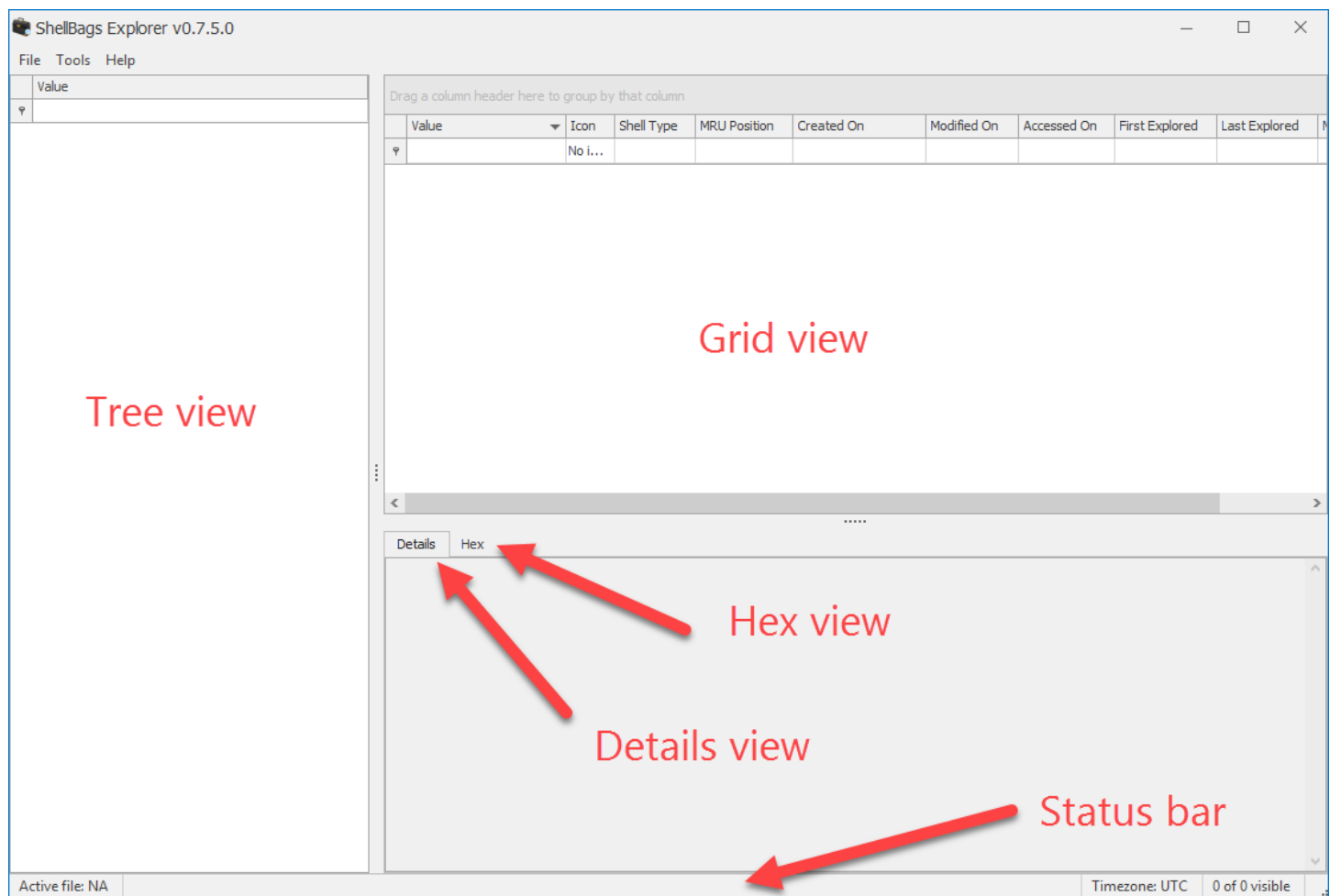
- ShellBagsExplorer.exe: The GUI version of ShellBags Explorer
- ShellBagsExplorerManual.pdf: The file you are reading now

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. When viewing nodes recursively, another icon (blue arrow) will be displayed to indicate this.

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 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

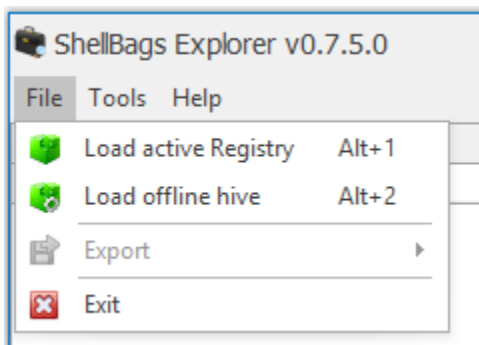
After a ShellBag is selected, the hex view will contain the raw hex content of the selected ShellBag. In the lower right of the hex display is a button which, when clicked, will display a data interpreter that is updated in real time as data is selected in the hex editor. This allows for verification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

Status bar

The status bar on the bottom contains various details about the loaded hive and whether a filter is active (including the number of visible rows vs. the total).

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 0.8.0.0, 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.

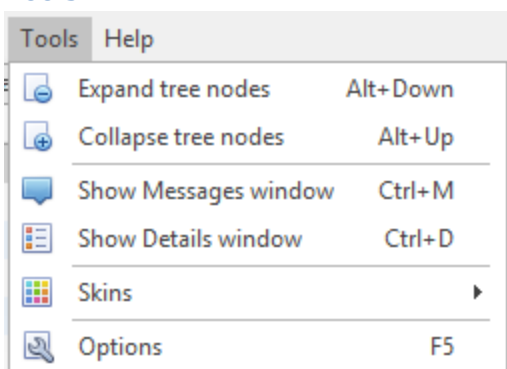
Export submenu

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 exactly they are shown in the grid.
- json: All data is exported in json format.

NOTE: You can customize the columns shown in the grid view by right clicking on any column header 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 and Details windows, or set other options such as the time zone to be used when displaying dates and times. The Messages window is displayed automatically as messages are generated.

Details

The Details window displays all available columns from the grid whether they are hidden or not. This allows for customization of the main grid while still having access to all of the underlying column data.

Details	
Value	usb-over-ethernet.zip
Icon	
Shell Type	File
Bag Path	BagMRU\1\2\1
Absolute Path	Desktop\My Computer\p:\Security Tools\usb-over-ethernet.zip
Type Indicator	0x32
Slot	3
MRU Position	0
Child Bags	0
Node Slot	100
Created On	2012-03-20 17:55:44
Modified On	2012-03-20 17:54:38
Accessed On	2012-03-20 17:55:44
LastWrite Time	2012-03-20 18:00:13
First Explored	2012-03-20 18:00:13
Last Explored	2012-03-20 18:00:13
MFT Entry	27686
MFT Sequence Number	292
Miscellaneous	NTFS file system
Extension Blocks	Beef0004
Extension Block Count	1
Always on top <input type="checkbox"/>	

Options

Options

☐ Show parent tree nodes when filtering

☒ Show horizontal scrollbar on grid

DateTime format: [Click for examples](#)
 Preview: 2017-01-17 16:26:12

Default timezone:

☒ Automatically submit unknown GUIDs and Shell IDs

Save Cancel

These options do the following:

Show parent tree nodes when filtering: When checked, parent nodes in the tree are shown as well as any matching nodes based on the filter. When unchecked, only nodes matching the filter will be shown.

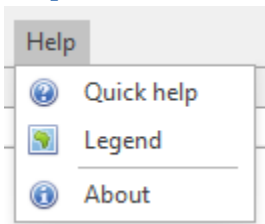
Show horizontal scrollbar on grid: When checked, a scrollbar will be shown on the grid (when necessary) to allow for the contents of all columns in the grid to be displayed completely. When unchecked, no scrollbar will be shown and the columns will be auto sized to fit the available data. This can lead to some data not being shown entirely without column resizing.

DateTime format: The default format for all dates and times throughout the program.

Default time zone: The time zone to use when displaying timestamps. The time zone should be selected before loading Registry data. UTC is the default.

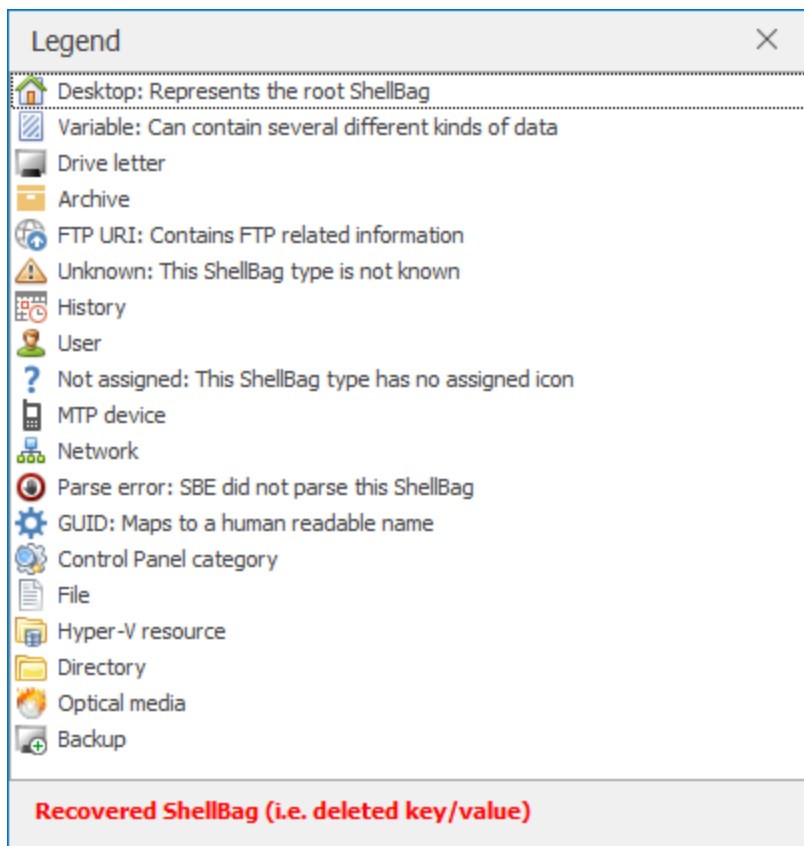
Automatically submit unknown GUIDs and Shell IDs: When checked, any unknown GUIDs or Shell IDs will be automatically submitted to Eric Zimmerman to add support for the unknown items. No identifiable information is sent other than the GUID or the byte array identifying the Shell item. It is highly recommended that this option remain enabled in order to ensure new data is discovered and support is added for unknown items.

Help



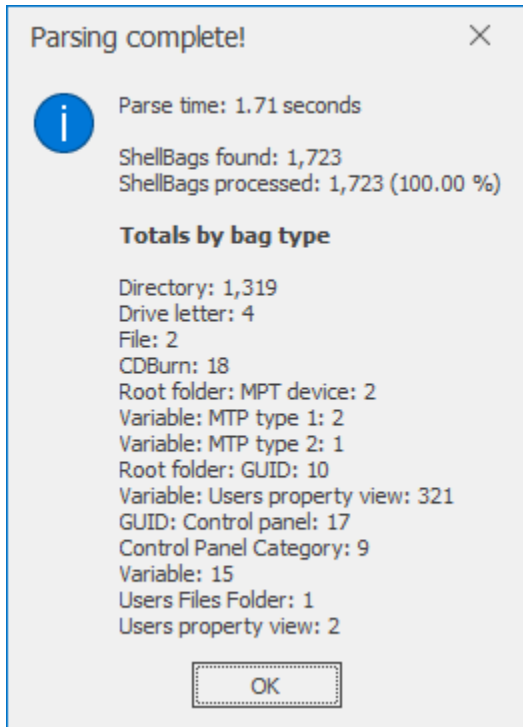
Quick help provides basic information on how to use SBE. About contains version information and contact information for the developer.

The Legend displays a list of icons used in SBE along with their meanings.



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.



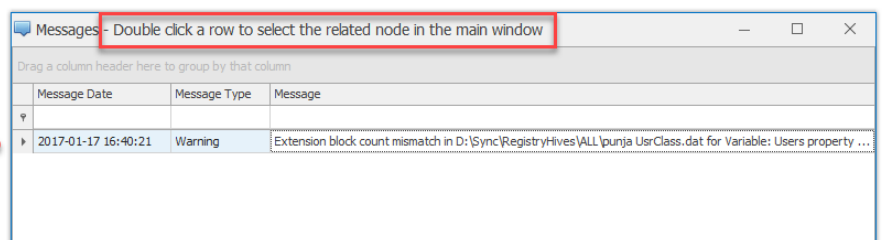
In the image above, notice 1,723 ShellBags were found in the Registry and 1,723 ShellBags were processed. Should there be a situation where the number of ShellBags found is not equal to the number of ShellBags displayed by SBE, the summary screen will be different to reflect this fact.

Messages window

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

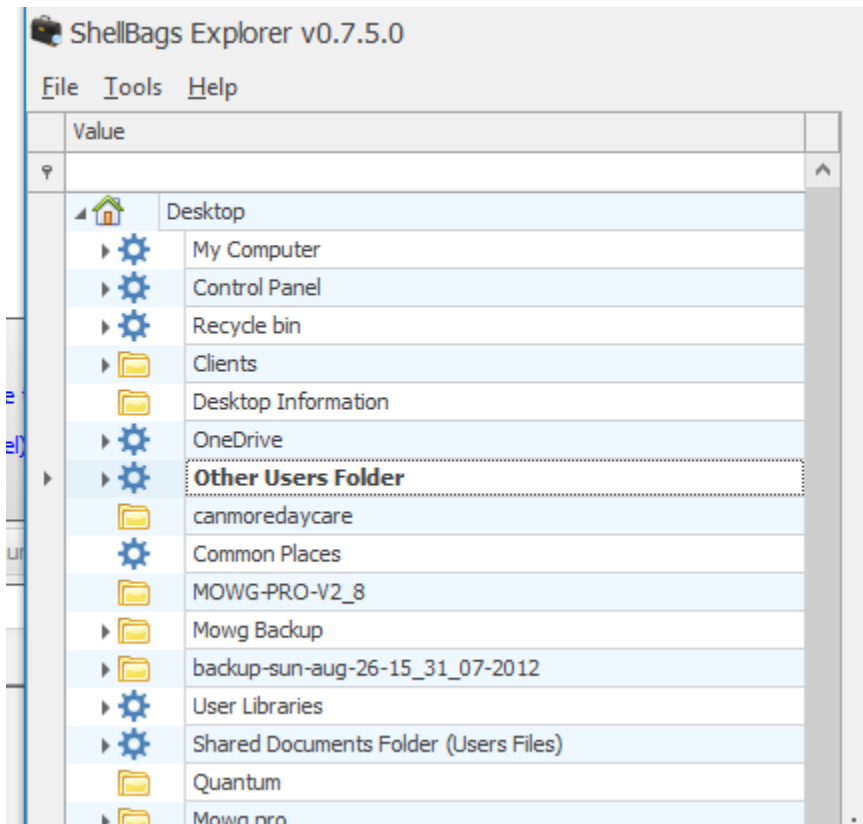
The Messages window can contain both errors and informational messages. Informational messages will be displayed when unknown GUIDs are found, new extension blocks are discovered, or there is a mismatch between the number of extension blocks in a ShellBag and the number of parsed extension blocks (these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

A screenshot of a Windows File Explorer window. The address bar shows the path 'This PC > Computer > KATHY-PC'. The main pane displays a list of files and folders. A red arrow points to a file named 'True::BLOB data: 4D-03-14-00-1F-50-E0-4F-D...'. Other visible items include 'OneDrive', 'Other Users Folder', 'kathy', 'KATHY-PC', 'credacycare', 'Common Places', 'MOWG-PRO-V2_8', 'Mowg Backup', 'backup-sun-aug-26-15_31_07-2012', 'User Libraries', 'Shared Documents Folder (Users Files)', 'Quantum', and 'Mowg pro'.

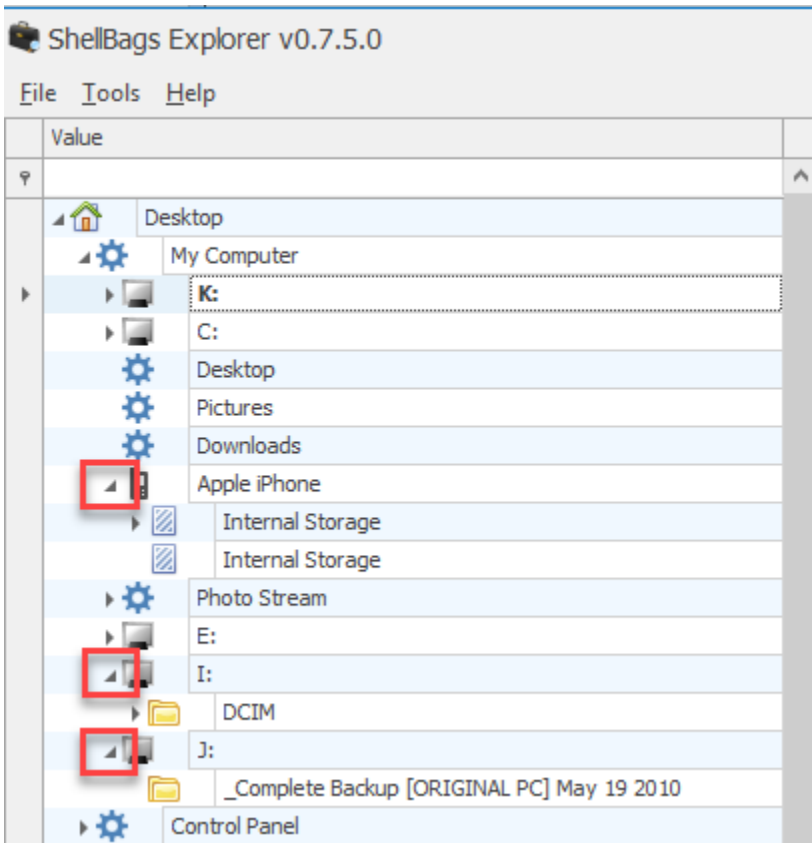


Tree view

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

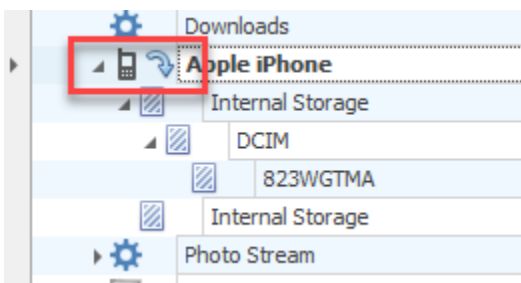


Expanding a node displays its children. In the image below, several child objects have been expanded.



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. The Legend contains a list of all available icons.

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.



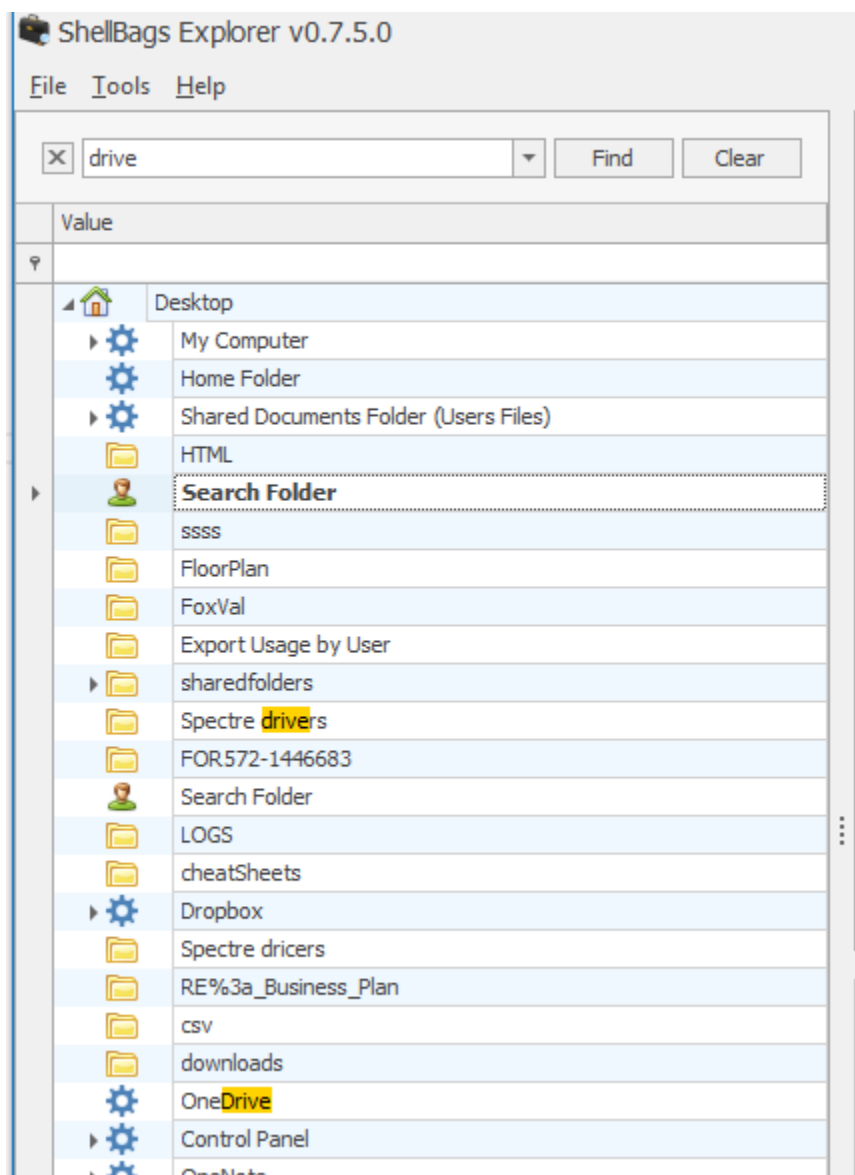
In the example above, Apple iPhone has been recursively explored. Notice the icon now shows a blue arrow indicating the node is being explored recursively. Exploring recursively lets you drill down into specific areas of interest and, as we will soon see, easily search for things across a given set of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

Finding data in the tree

There are several ways to quickly locate nodes in the tree.

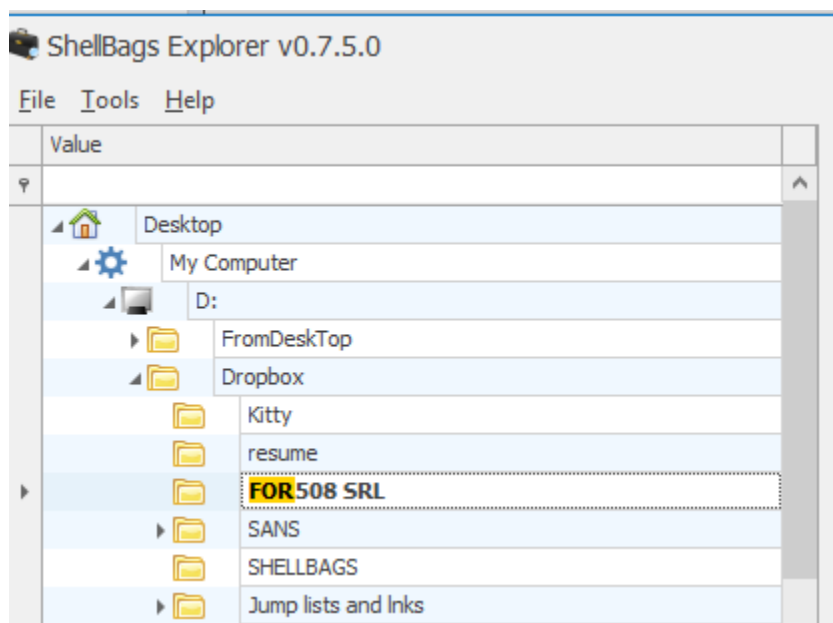
Find panel

Pressing CTRL-F will bring up the Find panel. Entering a search term in the find box will highlight any matching nodes, as shown below.



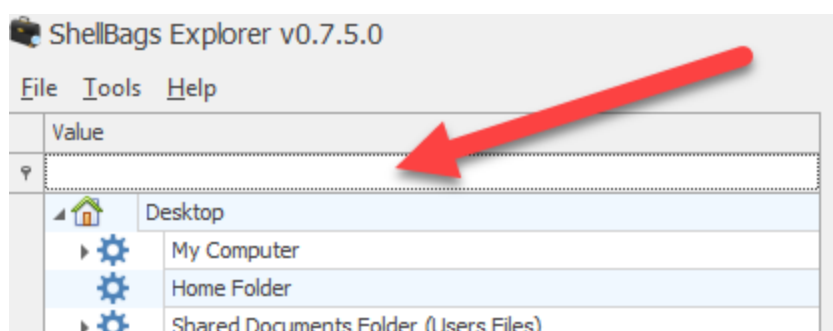
Typing on nodes

With a node selected, typing the name of a node will automatically initiate a search and expand the nodes as needed as matches are found. In the example below, the string “for” was typed with the Desktop node initially selected.



Filter

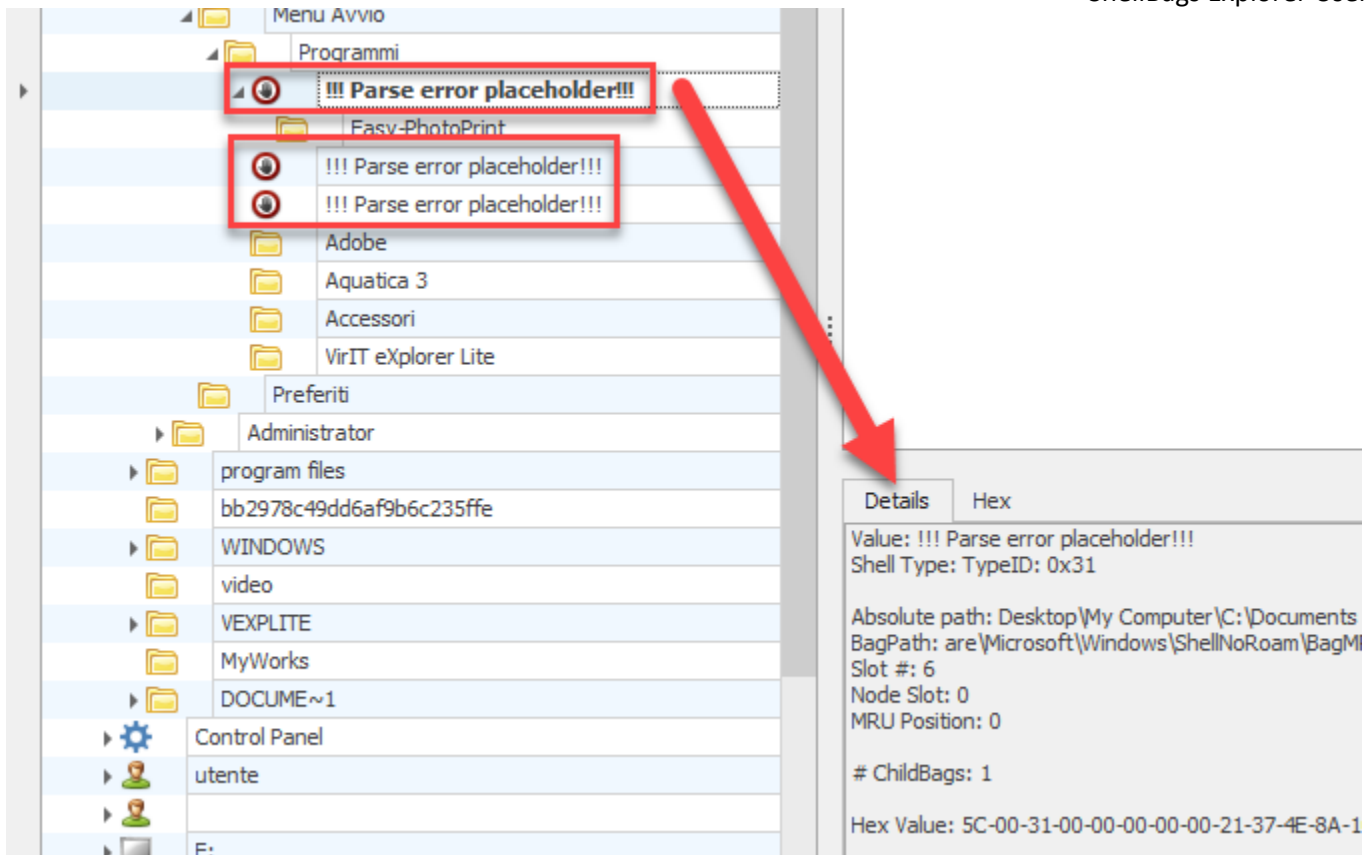
As with the grid, entering a filter criteria below the column header will filter nodes based on what is entered.



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.



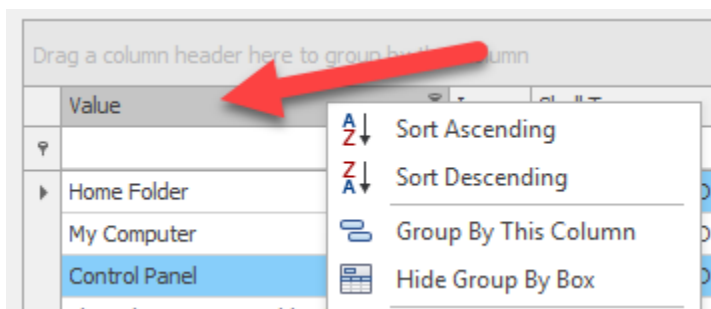
Grid view

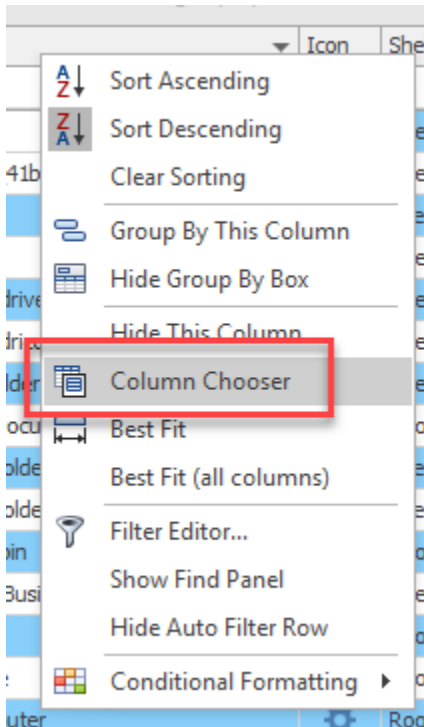
As nodes are selected in the tree view, the grid view is updated to show the child ShellBags of the selected node. In the case of recursive viewing, all child bags from the selected node and all child nodes are displayed in the grid view.

The grid view serves to view Shellbags in a consistent manner. Not all bags have data for all 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.

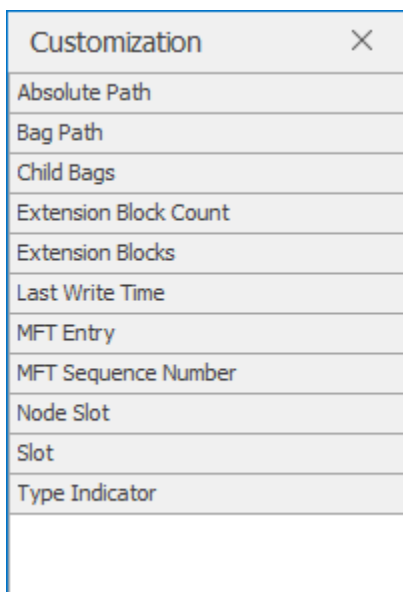
Customizing grid view columns

Right clicking on a column header of the grid view allows for customizing the fields displayed in the grid.





The column chooser is then shown and columns can be dragged to and from the main grid.



Column definitions

SBE supports the following columns as of v0.8.0.0:

- Absolute path: The absolute path from the Desktop to the location of the shellbag
- 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)

- Extension block count: The total number of extension blocks found in this shellbag
- Extension blocks: The names of unique extension blocks found in the shellbag
- First Explored: When available, the timestamp a folder was first explored
- Icon: A visual identifier for the shellbag
- 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 Sequence number
- Miscellaneous: Used to report additional items of interest about shellbags 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 Position: The Most Recently Used position of the bag
- Node Slot: Used to tie the shellbag to its counterpart in the Bags key
- Slot: The shellbag's numerical identifier in its parent key
- Shell Type: The human readable description of what kind of data the shellbag represents (file, folder, etc.)
- Type Indicator: The hexadecimal identifier for the shellbag. This is found in offset 0x02 in the hex that makes up a shellbag.
- Value: The name of the file, folder, property view, etc. for the shellbag. This is the primary identifier for a shellbag.

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

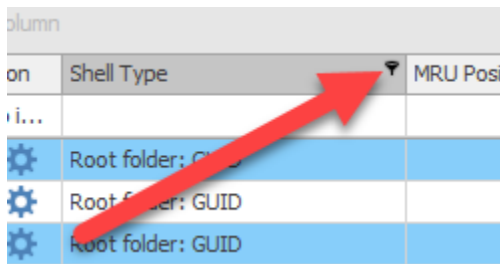
At the top of each column is a filter cell. In the image below it is highlighted in yellow.

Drag a column header here to group by that column

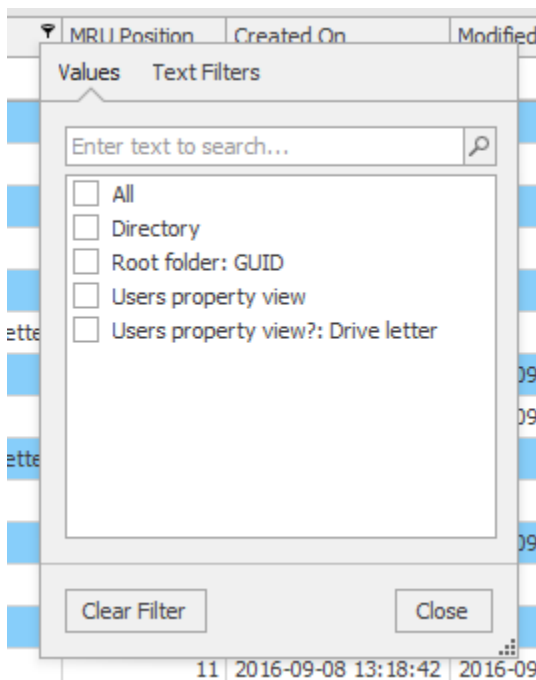
	Value	Icon	Shell Type	MRU Position	Created On	Modified On	A
▼		No i...					
▶	Home Folder	⚙️	Root folder: GUID	1			

Clicking on this cell and typing will immediately filter that column to only items containing the entered text.

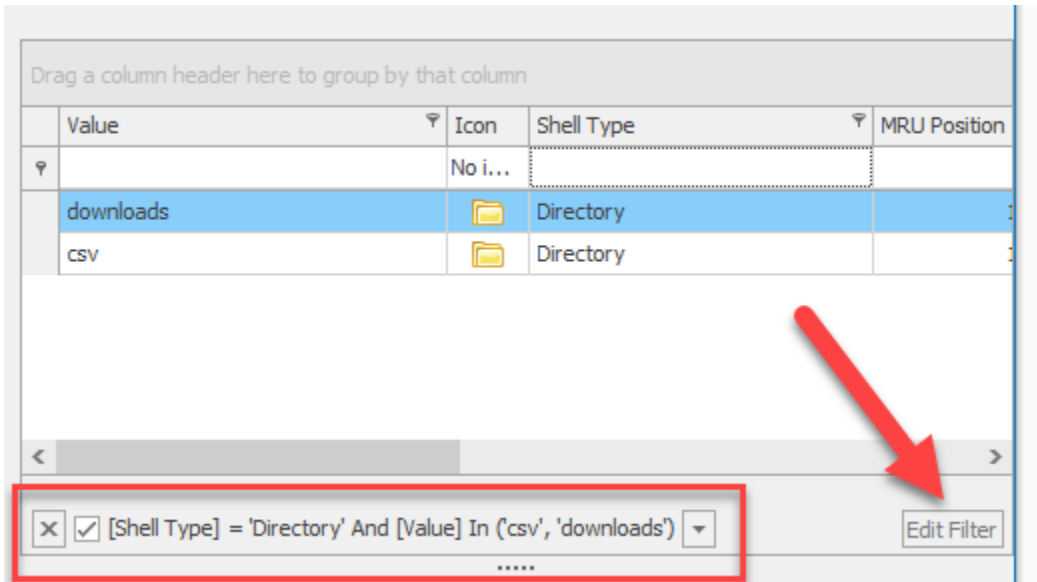
There are also dropdowns which contain all unique entries in the column plus options to add custom filters to the list.



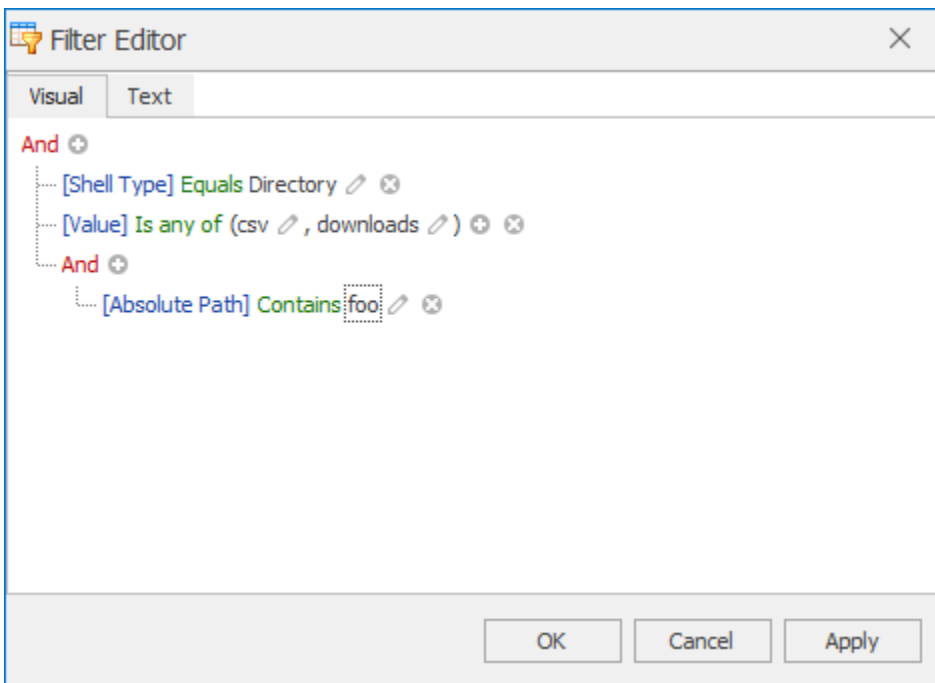
Clicking the funnel icon brings up an Excel style filter, as shown below.



When a filter is in place, the filter criteria will be displayed below the grid.



Clicking the X on the left removes all filters. Unchecking the box temporarily disables the filter. To the right is a button that allows for editing the filter criteria for more advanced requirements.













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.

Shell Type ▲						
	Value	Icon	MRU Position	Created On	Modified On	Access
▼		No i...				
▶	Shell Type: Directory					
▶	Shell Type: Root folder: GUID					
▶	Shell Type: Users property view					
▶	Shell Type: Users property view?: Drive letter					

In the example above, the “Shell Type” column was grouped. Clicking on the arrow expands a group.

Shell Type ▲			
	Value	Icon	MRU F
▼		No i...	
▶	Shell Type: Control Panel Category		
▶	Shell Type: Directory		
▶	Shell Type: Drive letter		
	D:		
	C:		
	E:		
	F:		
	I:		
	L:		
	S:		
	W:		
	M:		
	U:		







Clicking the arrow again 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.

Shell Type ▲

MRU Position ▲

	Value	Icon	Created
♀		No i...	
▶	▶ Shell Type: Control Panel Category		
	▲ Shell Type: Directory		
	▶ MRU Position: 0		
	▲ MRU Position: 1		
	20160831134705_Live Response data for WORKSTATION_95a65652-63b6-4a24-8764-b9785e...		2016-08
	Encryption		2015-11
	ReportFiles		2015-11
	100APPLE		2016-11
	Iphone Pics		2016-11
	WMTParser		2016-10

In the example above, we first grouped by Shell Type, then by MRU Position. 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). Right clicking a shellbag in the grid will select that shellbag in the tree.

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

The screenshot shows the ShellBags Explorer v0.7.5.0 interface. The left pane displays a tree view of the file system, with the C: drive selected. The right pane shows a list of bags (folders) under the selected drive, including TY15, TurboTax, TTWin, Toolbar History, tl, thumbnails, test, Temp, Tasks, System32, and Sync. The 'test' bag is highlighted. Below the list, the 'Details' tab is active, showing the following information:

Details	Hex
Value: C:	
Shell Type: Drive letter	
Bag Path: BagMRU\1, Slot #: 1, MRU Position: 1, Node Slot: 7	
Absolute Path: Desktop\My Computer\C:	

Notice the C: drive is highlighted in the tree view and the details view reflects this. If the bag with value “test” is right clicked, notice the tree view is updated and the “test” bag is highlighted and the grid view is updated to show the child bags of the test bag.

ShellBags Explorer v0.7.5.0

File Tools Help

Value	Icon	Shell Type	MRU Position
Value	No i...		
xp-tdungan-c-drive	Folder icon	Directory	

Drag a column header here to group by that column

Value	Icon	Shell Type	MRU Position
Value	No i...		
xp-tdungan-c-drive	Folder icon	Directory	

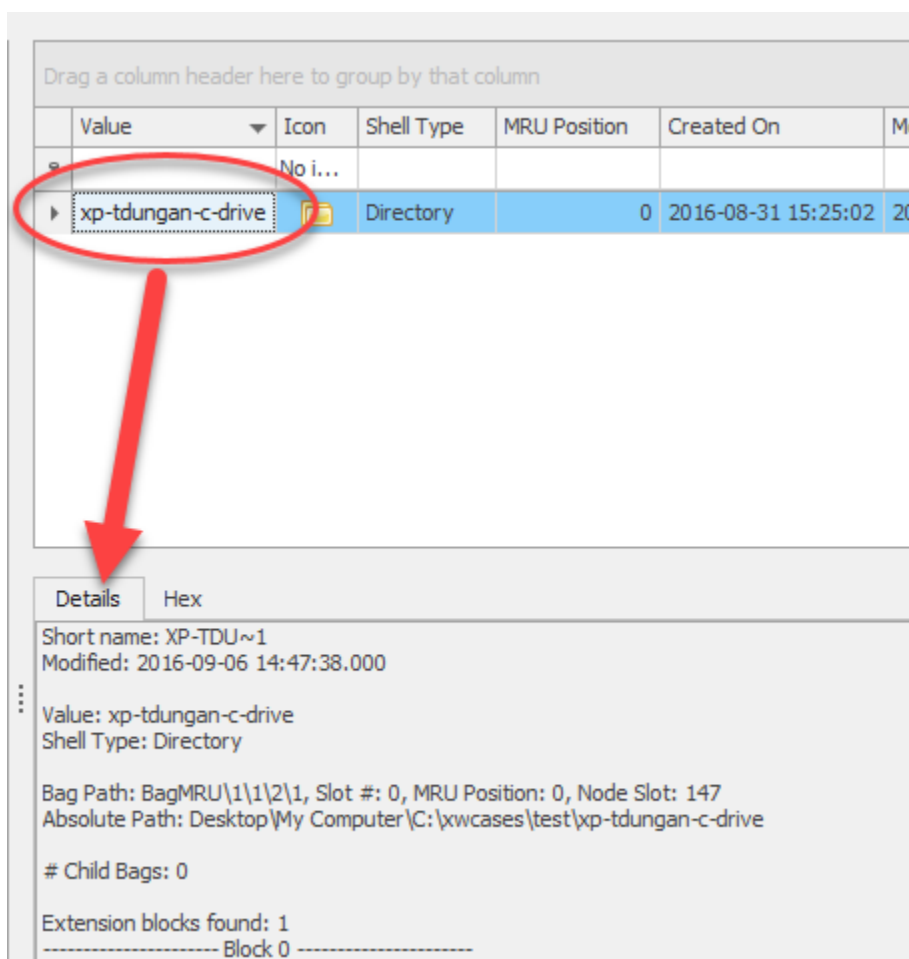
Details Hex

Short name: XP-TDU~1
Modified: 2016-09-06 14:47:38.000

Value: xp-tdungan-c-drive
Shell Type: Directory

Bag Path: BagMRU\1\1\2\1, Slot #: 0, MRU Position: 0, Node 5
Absolute Path: Desktop\My Computer\C:\xwcases\test\xp-tdu

The details pane is also updated with the newly selected bag in the tree view. Clicking the “xp-tdungan-c-drive” bag in grid view will update the details view to the “xp-tdungan-c-drive” 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 (<https://goo.gl/6dl4FS>)):

- 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.

A unique signature identifies extension blocks, 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 must 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	.. ADM NL~1..
48	00	09	00	04	00	EF	BE	43	45	EC	72	H. . . . T 3/CE1 r
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 c . . 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

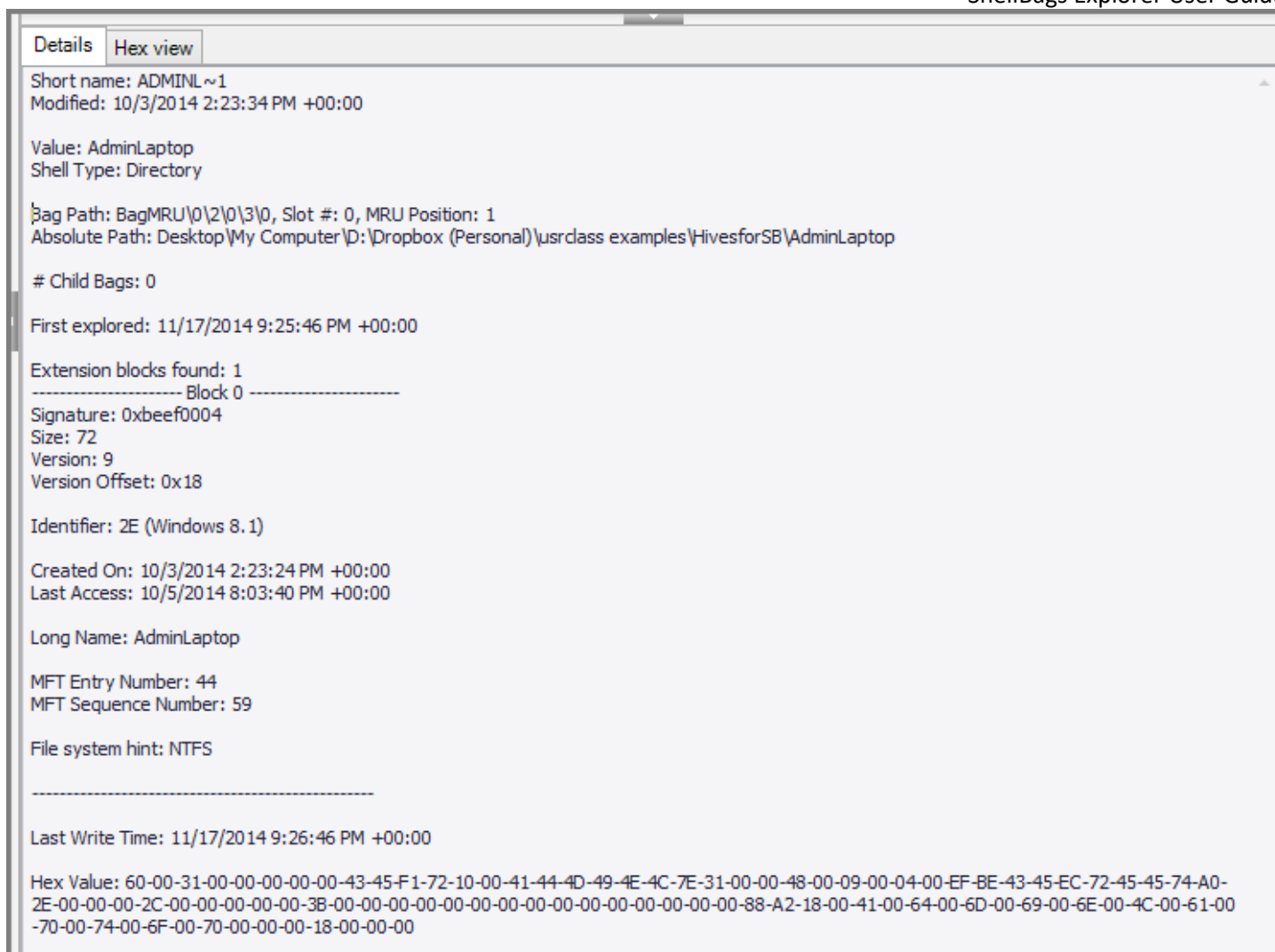
In the image above, we can see:

- 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 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 four 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.

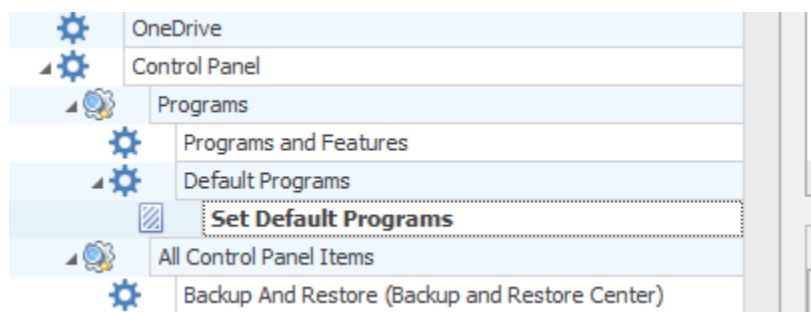


Here we can see the short name, MAC dates and times, full bag path, the absolute path, and the first explored date.

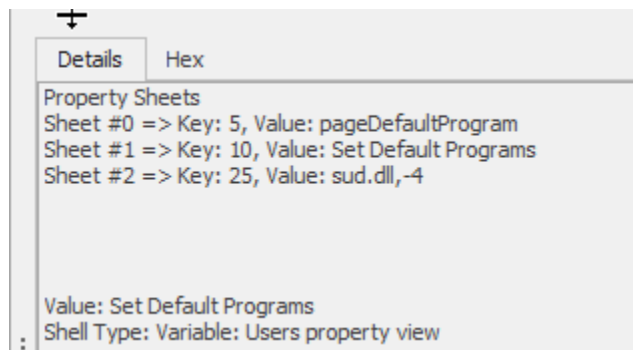
Note: The Last Write Time of the PARENT key (BagMRU\0\2\0\3\0) this ShellBag lives in is 11/17/2014 9:26:46 PM +00:00, but the First explored timestamp is 11/17/2014 9:25:46 PM +00:00. This is because First explored is determined by the last write date of the “bottom” key corresponding to BagMRU\0\2\0\3\0, 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.

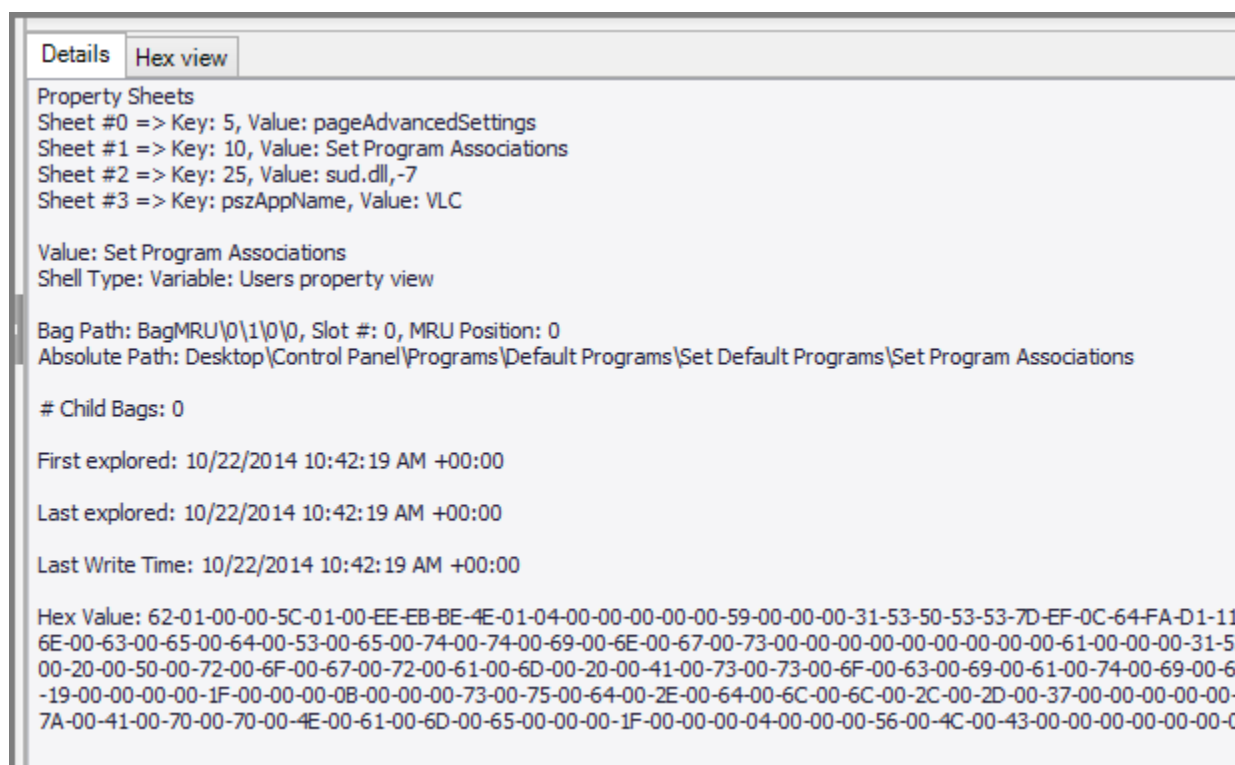
Some ShellBags contain embedded property sheets. These are another data structure commonly seen in various Windows artifacts and essentially are key/value pairs. In the example below, the bags related to the Control Panel have been expanded.



After selecting the “Set Default Programs” bag, its details view contains:



To illustrate this in another way, consider the example below.

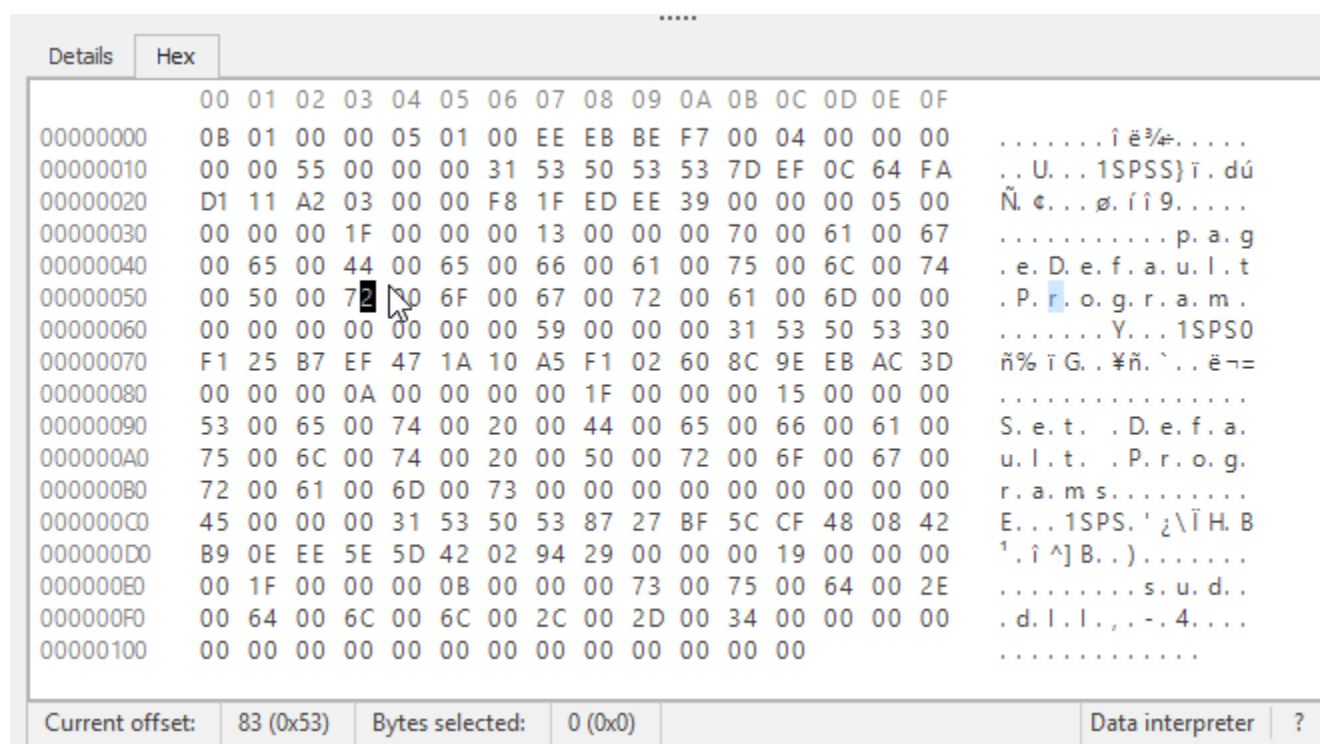


Here we can see the various property sheets and the data contained therein. Notice that now it is possible to tell not only WHEN program associations were changed, but WHAT program was used to take over those associations.

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. In the lower right is a button that brings up a Data interpreter.

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.

Data Interpreter

Numbers

8 bit, signed	103
8 bit, unsigned	103
16 bit, signed	103
16 bit, unsigned	103
32 bit, signed	7,471,207
32 bit, unsigned	7,471,207
64 bit, signed	30,681,189,080,760,423
64 bit, unsigned	30,681,189,080,760,423
Float	1.046939E-38
Double	1.29060870586473E-306

Dates and times

DOS FAT Time/date (32 bit)	3/18/1980 12:03:14 AM
DOS FAT Date/time (32 bit)	3/7/1980 12:03:36 AM
Windows FILETIME (64 bit)	3/23/1698 3:15:08 PM +00:00
Unix/Posix (32 bit)	3/28/1970 11:20:07 AM +00:00

Other

GUID	00720067-0061-006d-0000-000000000000
Maps to	
IP Address	103.0.114.0

Strings

ASCII	g
Unicode	gram
Base64	ZwByAGEAbQAAAAAAAAAAAFkAAAAxU1BTMPeIt+9HGh...

NOTE: Data is interpreted from the current offset and is not based on the selected bytes

Offset: **87 (0x57)** Always on top ☐ ?

Current offset: 87 (0x57) Bytes selected: 0 (0x0) Data interpreter ?

In this example the cursor is at offset 0x57 and we see the Data interpreter showing different interpretations of the data as numbers, strings, IP addresses, etc.

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

Data Interpreter

Numbers	
8 bit, signed	104
8 bit, unsigned	104
16 bit, signed	1,640
16 bit, unsigned	1,640
32 bit, signed	653,133,416
32 bit, unsigned	653,133,416
64 bit, signed	4,960,609,480,035,141,224
64 bit, unsigned	4,960,609,480,035,141,224
Float	1.65163E-15
Double	4.46266558765985E+23

Dates and times	
DOS FAT Time/date (32 bit)	7/14/1999 12:51:16 AM
DOS FAT Date/time (32 bit)	3/8/1983 4:55:28 AM
Windows FILETIME (64 bit)	n/a
Unix/Posix (32 bit)	9/12/1990 9:56:56 AM +00:00

Other	
GUID	26ee0668-a00a-44d7-9371-beb064c98683
Maps to	Control Panel
IP Address	104.6.238.38

Strings	
ASCII	h0x
Unicode	A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0
Base64	aAbuJgqg10STcb6wZMmGgwAA

NOTE: Data is interpreted from the current offset and is not based on the selected bytes

Offset: **4 (0x4)** Always on top ☐ ?

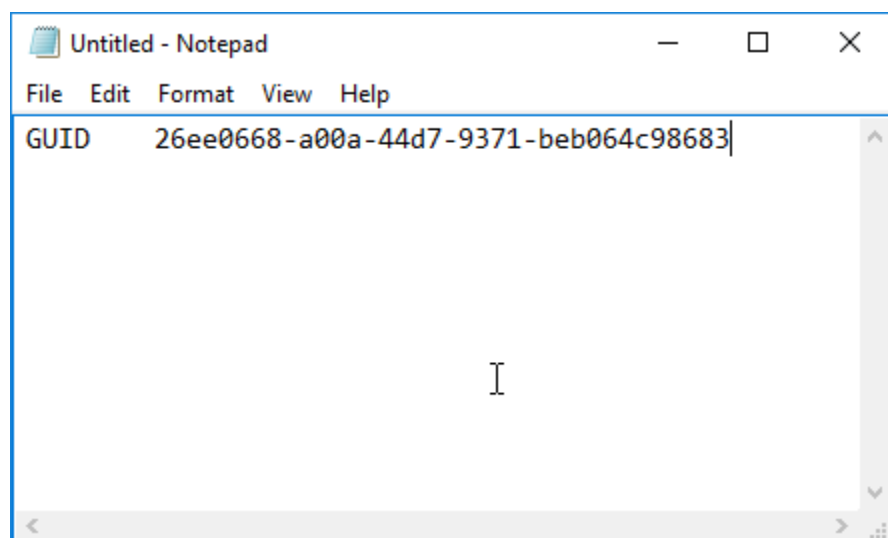
Current offset: 4 (0x4) Bytes selected: 0

Hex dump (offset 00-07):

00	01	02	03	04	05	06	07
14	00	15	70	68	06	EE	26
64	C8	86	83	00	00		

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

Finally, the data from the Data interpreter and its corresponding ShellBag details can be copied to the clipboard by selecting the data to copy and pressing CTRL-C. An example is shown below of what this data may look like once copied into a text editor.



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.

```

λ SBECmd.exe -?

SBECmd version 0.8.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman

    d      Directory to look for Registry hives. Required
    csv     Directory to save output to. Required

    dedupe  When true, SBECmd processes all hives in -d <directory> and removes duplicates. See manual for details

    dt      Date/time format string to use. Default is 'yyyy-MM-dd HH:mm:ss'
    tz      Timezone to use (Default = UTC). Enclose in quotes. Use '--tz list' for options

Examples: SBECmd.exe -d c:\temp\hives --csv c:\temp\sbeout
          SBECmd.exe -d c:\temp\hives --csv c:\temp\sbeout --tz "US Eastern Standard Time"
          SBECmd.exe -d c:\temp\hives --csv c:\temp\sbeout --dedupe

    Short options (single letter) are prefixed with a single dash. Long commands are prefixed with two dashes
  
```

Processing multiple, unrelated hives

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

```
SBECmd.exe -d <directory> --csv <output directory>
```

The directory specified by --csv will contain the TSV reports generated by SBECmd.exe. An example run is shown below.

```

SBECmd.exe version 0.8.0.0
Directory to process: c:\Temp\sss
Export format: TSV
All messages will be saved to 'c:\Temp\sss\!SBECmd_Messages.txt'
Deduplication: False
Timezone: UTC

Processing 'c:\Temp\sss\1UsrClass.dat'
Header length is smaller than the size of the file.
Found hbin with size 0 at absolute offset 0x121000
Initial processing complete. Building tree...
Found root node! Getting subkeys...
Hive processing complete!
Associating deleted keys and values...
Flushing record lists...

Parse time: 0.24 seconds

    Total Shell Bags found: 77

    Totals by bag type
    GUID: Control panel: 7
    Control Panel Category: 3
    Variable: Users property view: 7
    Root folder: GUID: 7
    Directory: 33
    Drive letter: 5
    Variable: 9
    File: 3
    Users Files Folder: 1
    Users property view: 1
    Network location: 1

Finished processing 'c:\Temp\sss\1UsrClass.dat'
Exported to: 'c:\Temp\sss\1UsrClass.dat.tsv'

-----
66-x64-AIOS-ENU.exe  64.5 MB  Application  7/25/2016 11:22 AM  -a-
Processing complete!
Processed 1 file(s) in 0.24 seconds!
Total Shell Bags found: 77

```

If SBECmd.exe encounters any errors processing hives, they will be reported to the console and to '!SBECmd_Messages.txt' in the directory specified by --csv.

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> --csv <out directory> --dedupe
```

The directory specified by --csv will contain the TSV report named 'Deduplicated.tsv'. An example run is shown below.

```

λ SBECmd.exe -d c:\Temp\sss\ForDedupe\ --csv c:\Temp\sss\ForDedupe\ --dedupe
SBECmd.exe version 0.8.0.0

Directory to process: c:\Temp\sss\ForDedupe\
Export format: TSV
All messages will be saved to 'c:\Temp\sss\ForDedupe\!SBECmd_Messages.txt'
Deduplication: True
Timezone: UTC

Processing 'c:\Temp\sss\ForDedupe\UsrClass 1.dat'
Header length is smaller than the size of the file.
Initial processing complete. Building tree...
Found root node! Getting subkeys...
Hive processing complete!
Associating deleted keys and values...
When getting values for nk record at absolute offset 0x2D1080, VK record at rel
nk record. Skipping!
When getting values for nk record at absolute offset 0x2D1080, VK record at rel
nk record. Skipping!
When getting values for nk record at absolute offset 0x3EC0F8, VK record at rel
nk record. Skipping!
When getting values for nk record at absolute offset 0x562088, not enough/inval
Value recovery is not possible
Error while processing deleted nk record at absolute offset 0x56FD38
Error while processing deleted nk record at absolute offset 0x575780
Flushing record lists...

-----

Processing 'c:\Temp\sss\ForDedupe\UsrClass 2.dat'
Header length is smaller than the size of the file.
Initial processing complete. Building tree...
Found root node! Getting subkeys...
Hive processing complete!
Associating deleted keys and values...
When getting values for nk record at absolute offset 0x5182E8, VK record at rel
record. Skipping!
When getting values for nk record at absolute offset 0x518810, VK record at rel
record. Skipping!
When getting values for nk record at absolute offset 0x51B5C8, not enough/inval
Value recovery is not possible
When getting values for nk record at absolute offset 0x52CE60, not enough/inval
alue recovery is not possible
When getting values for nk record at absolute offset 0x533618, VK record at rel
nk record. Skipping!
When getting values for nk record at absolute offset 0x533908, VK record at rel
nk record. Skipping!
When getting values for nk record at absolute offset 0x533908, VK record at rel
nk record. Skipping!
Error while processing deleted nk record at absolute offset 0x536968
Flushing record lists...

-----

Unique Shell Bags found: 162 (19.40 % duplicates)
Results saved to 'c:\Temp\sss\ForDedupe\Deduplicated.tsv'

Processing complete!
Processed 2 file(s) in 0.78 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 --tz switch. Enclose the time zone in quotes. An example is shown below. To get a list of all available time zones, use '--tz list'

```

λ SBECmd.exe -d c:\Temp\sss --csv c:\Temp\sss --tz "Mountain standard time"
SBECmd.exe version 0.8.0.0

Directory to process: c:\Temp\sss
Export format: TSV
All messages will be saved to 'c:\Temp\sss\!SBECmd_Messages.txt'
Deduplication: False
Timezone: Mountain standard time

Processing 'c:\Temp\sss\1UsrClass.dat'
Header length is smaller than the size of the file.
Found hbin with size 0 at absolute offset 0x121000
Initial processing complete. Building tree...
Found root node! Getting subkeys...
Hive processing complete!
Associating deleted keys and values...
Flushing record lists...

Parse time: 0.18 seconds

```

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

Cens		Editing	
	O	P	
Count	FirstExplored	LastExplored	Misc
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 0.8.0.0

NEW: This is a complete rewrite using all new controls including newer and more capable tree and grid. Allows for better filtering, grouping, etc. With this change, SBE has the same look and feel of Registry Explorer

NEW: Updated hex viewer and data interpreter

NEW: Updated icons throughout

NEW: Added Legend to Help menu

NEW: Added Options dialog

NEW: Added skinning

NEW: Completely updated manual to reflect new version

NEW: Added new GUIDs and new extension block types

NEW: Added ability to automatically report unknown GUIDs, shell IDs and extension blocks for reversing and inclusion in future releases

NEW: Added Details form which allows for viewing all available details regardless of the columns shown in grid view

CHANGE: Time zone moved to Options dialog

CHANGE: SBECmd internals reworked to simplify logging and make command line options consistent with other software

FIX: Added CD file system indicator vs showing FAT for optical media

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 shellbags

FIX: Handle Unicode better in directory shellbags

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/libfws/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>