The Forensic Value of the Windows 7 Jump List

Alexander G. Barnett

Purdue University agbarnet@purdue.edu

Abstract. The Windows 7 Jump List is an aspect of the Windows 7 operating system that has the potential to contain data and artifacts of great interest to investigators, but has yet to receive any considerable attention or research. As of this writing, only one published work makes mention of their existence, and no tools exist to automate their retrieval and analysis. The goal of this research is to provide an overview of the function and behavior of jump lists, and also to examine the structure of jump lists with the intention of proposing further research for making use of them in a forensic capacity.

Keywords: Jump List, Windows 7, Forensics.

1 Introduction

Windows Jump Lists have the potential to be an excellent source of evidence for investigators to collect, yet have not been the target of much academic scrutiny. Due to this lack of research, the intent of this paper is provide a basic understanding of the behaviors, data, and structures associated with Jump Lists, and to propose new avenues of research for their exploitation in a forensic capacity. The first section of this paper will provide an overview of Jump Lists and their end-user functions, while the second will document several experiments to determine how Jump Lists behave under certain circumstances. The third section will examine the internal structure of the Jump List, and the fourth and final section will recommend future research.

1.1 Overview

Jump Lists are a feature new to Windows 7. They serve a number of purposes, depending on the specific program utilizing the jump list. For example, the Microsoft Word Jump List contains a list of recently opened documents, as well as a section for user-defined "pinned" documents that never leave the list.

Jump Lists may also appear in the Start Menu, where they duplicate the functionality of taskbar Jump Lists.

The Microsoft Word Jump List demonstrates the basic function of Jump Lists, which is primarily to show files recently used by specific programs. Other programs, such as AOL Instant Messenger, utilize their Jump Lists for additional tasks.

P. Gladyshev and M.K. Rogers (Eds.): ICDF2C 2011, LNICST 88, pp. 197-210, 2012.

[©] Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2012



Fig. 1. The Microsoft Word taskbar Jump List



Fig. 2. The Microsoft Word Jump List in the Start Menu

AIM's Jump List contains shortcuts for sending new instant messages, changing the user's online status, reading their mail, and changing the program's settings. While program shortcuts may be useful, it is the recent items feature that is of primary interest to investigators. These lists may contain files that have been created, downloaded, uploaded, or opened, depending on the program being used, and retain the items even after the actual file is deleted. What's more, the Jump List may record items the user does not know are being recorded (such as files downloaded while in Firefox private browsing mode), making them even more useful for investigators.



Fig. 3. The AOL Instant Messenger Jump List

Completely clearing a single Jump List is not an intuitive process. There are three methods the user can utilize: The first is to manually remove each item in the list by right clicking on the Jump List entry and selecting "Remove from this list", which is not practical for larger lists. The second method is to right click on the Start Menu icon and select Properties, select the Start Menu tab, and uncheck both options in the Privacy section, which clears all Jump Lists, not just one. The third method is to manually delete the Jump List file located at:

%APPDATA%\Microsoft\Windows\Recent\AutomaticDestinations or

$\% APPDATA\% \verb|Microsoft|\verb|Windows|\verb|Recent|CustomDestinations.$

This last item is of particular interest for a number of reasons. First, the *AutomaticDestinations* and *CustomDestinations* folders are hidden. This would normally not be an issue to an experienced user, however even after changing the Windows settings to show hidden files and folders, these particular folders remain hidden. This unusual behavior means that, unless the exact path of the folder is known, the user cannot access individual Jump List files. Additionally, even if the user were able to locate the Jump List files, deleting the correct list would be difficult given that Jump List file names are encoded in a seemingly random string of characters. Without viewing the contents of the file in a HEX editor as depicted in Appendix A, it is impossible to know which program the list represents. Due to the complexities involved with deleting Jump List data, the records they contain represent a good source of possible evidence regarding user activities for investigators.

2 Experiments

The following experiments were carried out on a PC running Windows 7 Ultimate (64-bit). The Jump List files were manually deleted from the *AutomaticDestinations* and *CustomDestinations* directories at the beginning and conclusion of each experiment. Experiments are grouped by the program being tested.

2.1 Firefox 3.6.16

Experiment #1: Downloading images in normal browsing mode.

For this experiment, images were downloaded from various websites by right-clicking on the image selecting "save-as".

Result: Images saved in this way appeared in the Firefox Jump List after downloading completed.

Experiment #2: Downloading images in private browsing mode.

This experiment mimics experiment #1, with the exception of placing the browser in private browsing mode before beginning.

Result: Images saved appeared in the Firefox Jump List and remained there after the program was closed. These images did not appear in the Firefox disk cache, which was viewed through the about:cache Firefox interface.

Experiment #3: Uploading images in normal browsing mode.

For this experiment, images were uploaded to an online image board through the site's upload function.

Result: Uploaded images were listed in the Firefox Jump List. Interestingly, the images appeared in the Jump List immediately upon being confirmed in the file dialog box, not after being uploaded to the server.

Experiment #4: Uploading images in private browsing mode.

For this experiment, images were uploaded to an online image board through the site's upload function while Firefox was in private browsing mode.

Result: Uploaded images were listed in the Firefox Jump List. Again, the images appeared in the Jump List immediately upon being confirmed in the file dialog box, not after being uploaded to the server.

Experiment #5: Comparing the Jump Lists of uploaded and downloaded files.

For this experiment, an image was uploaded to an image board in normal browsing mode. After uploading, the Jump List file was copied to another folder and renamed upload.automaticDestinations-ms. After this, the original Jump List file was deleted and the same image downloaded from the image board. The new Jump List file was copied to another folder and renamed download.automaticDestinations-ms. The MD5 hash value of each file was then calculated.

Result: The hashes did not match. Viewing the HEX values of each file revealed substantial differences. This could indicate that Jump Lists have an internal mechanism for differentiating uploaded and downloaded files.

Experiment #6: Uploading a file to a flash-based website.

For this experiment, an image file was uploaded to a flash-based website through Firefox in normal browsing mode. The upload function also appeared to be flash-based, although it used a standard file browser dialog.

Result: The file did not appear in the Firefox Jump List.

	Experiment	Result
1.	Downloading an image in normal	Item appeared in the Jump List.
	browsing mode	
2.	Downloading an image in private	Item appeared in the Jump List.
	browsing mode.	
3.	Uploading an image in normal	Item appeared in the Jump List.
	browsing mode.	
4.	Uploading an image in private	Item appeared in the Jump List
	browsing mode.	
5.	Comparing Jump Lists containing	MD5 hash values did not match.
	one downloaded file and one	
	uploaded file.	
6.	Uploading a file to a flash-based	Item did not appear in the Jump List.
	website.	

Table 1. Summary of Firefox Results

2.2 Firefox – Conclusions

The most notable aspect of the Firefox Jump List's behavior is that files downloaded and uploaded, even in private browsing mode, are recorded. If the user overlooks this behavior, the Firefox Jump List could provide a telling log of activities performed online. Additionally, the fact that a Jump List containing only one item downloaded and a Jump list containing only one item uploaded are different reveals that Jump Lists may have some sort of mechanism for differentiating how the file was processed through the browser. This feature could become vitally important if, for example, a case wanted to prove that a user distributed a file rather than merely acquired it. Finally, the fact that an image uploaded through a flash-based interface does not appear in the Jump List reveals that Jump Lists do not record 100% of file uploads, and may in fact omit downloads and uploads from other methods as well.

2.3 Internet Explorer 8

The experiments performed on the Firefox browser were performed on Internet Explorer. For the sake of brevity, the procedures will not be re-listed.

Experiment #1: Downloading images in normal browsing mode.

Result: The file did not appear in the Internet Explorer Jump List in the Windows UI. However, viewing the *AutomaticDestinations* Jump List's HEX data revealed that the image was recorded.

Result 2: This test was repeated on a later date. A file was saved immediately after deleting the Jump List, after which the file was listed in the Jump List UI. However, after visiting several websites, the file disappeared from the list and was replaced by links to the websites recently visited.

Experiment #2: Downloading images in private browsing mode.

Result: The file did not appear in the Internet Explorer Jump List in the Windows UI. Also, it did not appear in the Jump List HEX data.

Experiment #3: Uploading images in normal browsing mode.

Result: The file did not appear in the Internet Explorer Jump List. However, viewing the *AutomaticDestinations* Jump List's HEX data revealed that the image was recorded.

Experiment #4: Uploading images in private browsing mode.

Result: The file did not appear in the Internet Explorer Jump List. Also, it did not appear in the Jump List HEX data.

Expirement #5: Comparing the Jump Lists of uploaded and downloaded files.

Result: The hashes did not match. Viewing the HEX values of each file revealed substantial differences.

Experiment	Result
1. Downloading an image in normal	Item did not appear in Jump List UI,
browsing mode	however it appeared in the Jump
	List HEX values.
2. Downloading an image in private	Item did not appear in the Jump
browsing mode.	List.
3. Uploading an image in normal	Item did not appear in Jump List UI,
browsing mode.	however it appeared in the Jump
	List HEX values.
4. Uploading an image in private	Item did not appear in the Jump
browsing mode.	List.
5. Comparing Jump Lists containing	MD5 hash values did not match.
one downloaded file and one	
uploaded file.	

Table 2. Summary of Internet Explorer Results

2.3.1 Internet Explorer – Conclusions

During the course of these experiments, it was noted that after visiting the same page several times, a link to the page was stored in the Internet Explorer Jump List. Also, these entries persisted after deleting the files located in the *AutomaticDestinations* directory, indicating the existence of a separate Jump List file. After researching the topic online, it was discovered that another set of Jump List files do indeed exist at %*APPDATA*%*MicrosoftWindowsRecent \CustomDestinations*. Purging these files removed the entries in the Internet Explorer Jump List UI, showing that a second Internet Explorer Jump List is stored in this location.

This discovery prompted the researcher to revisit experiments #2 and #4 (downloading and uploading images in private browsing mode) to see if the files were noted in the *CustomDestinations* Jump List. After re-performing the tests, it was noted that this second Jump List made no mention of the files either.

It was also interesting to note that the files uploaded and downloaded in private browsing mode were not stored in the Internet Explorer Jump List, while files uploaded and downloaded in private browsing mode using the Firefox browser were. This difference may indicate that Internet Explorer, being a core part of the operating system, has access to system methods inaccessible to other browsers.

It was noted that even after clearing both sets of Jump List files, upon restarting the browser, the Jump List referencing frequently visited sites was restored. However, after re-deleting the Jump List files, clearing the browser history, and restarting the browser, the Jump List was not repopulated. Based on this observation, it is reasonable to assume that this particular Jump List can be automatically generated from the browser's history files if deleted.

Finally, it was discovered through Experiment 1, Result 2 that the Internet Explorer Jump List will default to the *AutomaticDestinations* list if the *CustomDestinations* list is not available. However, it will revert back to the *CustomDestinations* list as soon as it is available.

2.4 Jump List File Names

Jump List file names, while appearing to be a random string of characters, always follow the format *16 characters dot automaticDestinations-ms* or *customDestinations-ms* (depending on which folder the file is present in). The 16 characters preceding the *.automaticDestinations-ms* or *.customDestinations-ms* are of particular interest to investigators, since they identify which program the list is associated with. The following tests will attempt to shed some light on how these lists are named.

Experiment #1: Are Jump List names static or dynamic?

The 16 characters appear to be random. In this experiment, several different programs' Jump Lists will be deleted and recreated multiple times to determine if the name is randomly generated.

Result: The file names did not change. Additionally, it was found that if a program had Jump Lists in both the *AutomaticDestinations* and *CustomDestinations* folders, the 16 character identifier was the same on both files. Table 3 lists the Jump List names of several programs:

Program Name	Jump List Name							
Firefox 3.6.16	5c450709f7ae4396							
Internet Explorer 8	28c8b86deab549a1							
Microsoft Word 2010	a7bd71699cd38d1c							
Windows Explorer	1b4dd67f29cb1962							
Notepad (64-bit)	9b9cdc69c1c24e2b							
Notepad (32-bit)	918e0ecb43d17e23							

Table 3. Jump List names

2.4.1 File Names – Conclusions

The fact that the identifiers are static and always sixteen characters long reveals that they are most likely encoded names of whichever program they represent. However, it is not clear at this time how these characters are encoded. Although they appear to be hexadecimal representations of characters (given that the characters stop at the letter f), translating the strings from HEX to ASCII text produces no meaningful results. It is possible that decoding the identifier would produce an eight-character DOS name, although this raises the question of how Windows can differentiate between programs with identical names (such as the 32 and 64 bit versions of Notepad).

2.5 File Structure

Examining the file structure of a Jump List is a difficult task. Viewing the file in plain text produces garbage text, so the only available method is to view the contents in a HEX editor. Even after viewing the file in HEX, making sense of the ensuing code is quite difficult. However, close examination reveals some commonalities in the structure of each Jump List. To begin, all Jump List files appear to begin with code depicted in figure 4:

Fig. 4. The first four lines of Jump List code

The next line is nearly identical in all files, with the exception of the first value which is always either "01" or "02". It is unknown at this time what this value signifies.

00000040 01 00 00 00 fe ff ff ff 00 00 00 00 00 00 00þÿÿÿ......

Fig. 5. The fifth line of code (Notepad 64 bit)

00000040 02 00 00 00 fe ff ff ff 00 00 00 00 00 00 00 00þÿÿÿ.....

Fig. 6. The fifth line of code (Firefox)

The next 27 lines contain the code shown in figure 7, after which the code is no longer uniform from list to list.

Fig. 7. Lines six through thirty-two

The ensuing structure is complex and seemingly repetitive. The complete file can be roughly represented as having the structure depicted in table 4. Figure 8 shows a partial view of the Firefox Jump List after uploading a file named cat.JPG to a website. There are several observations worth noting in the example. First, while the file path is shown three times in this particular figure, partial file paths are listed multiple times in the preceding code with large sections of unknown code in between, as shown in Figure 9. Also worth noting is the second file path, which contains the machine's host name (MAGUS in this case). Downloading a file produces a similar structure, with multiple repetitions of the file path along with host name. Comparing the structures of a Jump List with one uploaded file and a Jump List with one downloaded file reveals significant differences between the two files' structures, however there are no discernable sections of code which clearly indicate whether a file was uploaded or downloaded. Both lists contain references to the hostname and users@Shell32.dll, and both contain both full and partial file paths repeated many times throughout the file, although in seemingly different orders with small sections of illegible code interspaced. However, one immediately noticeable difference is that the Jump List containing the downloaded file is much longer (by approximately 2600 characters) than the Jump List containing the uploaded file. Although much of the data contained in a Jump List is incomprehensible at this time, investigators can still make use of the file paths they contain to prove that a file was used on that particular machine.

00001390	00	00	00	00	1f	00	00	00	28	00	00	00	43	00	3a	00	(C.:.
000013a0	5c	00	55	00	73	00	65	00	72	00	73	00	5c	00	41	00	\.U.s.e.r.s.\.A.
000013b0	6c	00	65	00	78	00	5c	00	50	00	69	00	63	00	74	00	l.e.x.\.P.i.c.t.
000013c0	75	00	72	00	65	00	73	00	5c	00	49	00	6e	00	74	00	u.r.e.s.\.I.n.t.
000013d0	65	00	72	00	6e	00	65	00	74	00	5c	00	63	00	61	00	e.r.n.e.t.\.c.a.
000013e0	74	00	2e	00	4a	00	50	00	47	00	00	00	25	00	00	00	tJ.P.G%
000013±0	03	00	00	00	00	1f	10	00	00	01	00	00	00	08	00	00	
00001400	00	70	00	69	00	63	00	74	00	75	00	72	00	65	00	00	.p.i.c.t.u.r.e
00001410	00	00	00	00	00	00	00	00	00	00	00	00	00	79	00	00	y
00001420	00	1c	00	00	00	03	00	00	00	1c	00	00	00	2d	00	00	
00001430	00	38	00	00	00	5a	00	00	00	11	00	00	00	03	00	00	.8Z
00001440	00	64	de	76	ac	10	00	00	00	00	43	3a	5c	55	73	65	.dÞv⊣C:\Use
00001450	72	73	5c	00	00	22	00	00	00	02	00	00	00	14	00	00	rs\"
00001460	00	00	00	00	00	00	00	02	00	5c	5c	4d	41	47	55	53	\\MAGUS
00001470	5c	55	73	65	72	73	00	41	6c	65	78	5c	50	69	63	74	\Users.Alex\Pict
00001480	75	72	65	73	5c	49	6e	74	65	72	6e	65	74	5c	63	61	ures\Internet\ca
00001490	74	2e	4a	50	47	00	39	00	00	00	09	00	00	aO	2d	00	t.JPG.9
000014a0	00	00	31	53	50	53	55	28	$_{\rm 4c}$	9f	79	9f	39	4b	a8	dO	1SPSU(LŸyŸ9K~Đ
000014b0	e1	d4	2d	e1	d5	£3	11	00	00	00	07	00	00	00	00	dO	áÔ-áÕó
000014c0	00	00	00	ff	ff	00	00	00	00	00	00	00	00	00	00	60	····ÿÿ······``
000014d0	00	00	00	03	00	00	aO	58	00	00	00	00	00	00	00	6d	
000014e0	61	67	75	73	00	00	00	00	00	00	00	00	00	00	00	72	agusr
000014f0	16	82	a6	56	00	d1	4e	96	a6	ec	18	cc	4a	d2	59	b1	., $V.\tilde{N}N i.IJOY\pm$
00001500	e6	bf	a6	2f	21	e0	11	8f	b9	00	1a	aO	8a	a9	c5	72	濦/!๊©År
00001510	16	82	a6	56	00	d1	4e	96	a6	ec	18	cc	4a	d2	59	b1	., $V.\tilde{N}N-$ i.ÌJÒY±
00001520	e6	bf	a6	2f	21	e0	11	8f	b9	00	1a	aO	8a	a9	c5	00	濦/!๊©Å.
00001530	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001540	01	00	00	00	01	00	00	00	00	00	00	00	00	00	80	3f	€?
00001550	01	00	00	00	00	00	00	00	01	00	00	00	00	00	00	00	
00001560	6e	55	fe	fb	30	06	6c	6d	72	16	82	a6	56	00	d1	4e	nUþû0.1mr.,¦V.ÑN
00001570	96	a6	ec	18	cc	4a	d2	59	b1	e6	bf	a6	2f	21	e0	11	-{ì.ÌJÒY±æ¿{/!à.
00001580	8f	b9	00	1a	aO	8a	a9	c5	72	16	82	a6	56	00	d1	4e	.'Š©År.,¦V.ÑN
00001590	96	a6	ec	18	cc	4a	d2	59	b1	e6	bf	a6	2f	21	e0	11	-¦ì.ÌJÒY±æ¿¦∕!à.
000015a0	8f	b9	00	1a	aO	8a	a9	c5	6d	61	67	75	73	00	00	00	. 'Š©Åmagus
000015b0	00	00	00	00	00	00	00	00	01	00	00	00	00	00	00	00	
000015c0	00	00	80	3f	cO	48	d2	4e	00	fb	cb	01	ff	ff	ff	ff	€?ÀHÒN.ûË.ÿÿÿÿ
000015d0	27	00	43	00	3a	00	5c	00	55	00	73	00	65	00	72	00	'.C.:.\.U.s.e.r.
000015e0	73	00	5c	00	41	00	6c	00	65	00	78	00	5c	00	50	00	s.\.A.l.e.x.\.P.
000015f0	69	00	63	00	74	00	75	00	72	00	65	00	73	00	5c	00	i.c.t.u.r.e.s.\.
00001600	49	00	6e	00	74	00	65	00	72	00	6e	00	65	00	74	00	I.n.t.e.r.n.e.t.
00001610	5c	00	63	00	61	00	74	00	2e	00	4a	00	50	00	47	00	\.c.a.tJ.P.G.

Fig. 8. Section of Firefox Jump List after uploading one file

00000d10	00	00	00	00	11	10	00	00	c1	01	00	00	14	00	1f	50	P
00000d20	e0	4f	d0	20	ea	3a	69	10	a2	d8	08	00	2b	30	30	9d	àOÐ ê:i.≎Ø+00.
00000d30	19	00	2f	43	3a	5c	00	00	00	00	00	00	00	00	00	00	. /C:\
00000d40	00	00	00	00	00	00	00	00	00	74	00	31	00	00	00	00	t.1
00000d50	00	30	3e	0f	23	11	00	55	73	65	72	73	00	60	00	08	.0>.#. Users.`
00000d60	00	04	00	ef	be	ee	3a	85	1a	30	3e	0f	23	2a	00	00	ĭ¾î:0>.#*
00000d70	00	22	02	00	00	00	00	01	00	00	00	00	00	00	00	00	. "
00000d80	00	36	00	00	00	00	00	55	00	73	00	65	00	72	00	73	.6U.s.e.r.s
00000d90	00	00	00	40	00	73	00	68	00	65	00	6c	00	6c	00	33	0.s.h.e.l.l.3
00000da0	00	32	00	2e	00	64	00	6c	00	6c	00	2c	00	2d	00	32	.2d.1.1.,2
00000db0	00	31	00	38	00	31	00	33	00	00	00	14	00	4a	00	31	.1.8.1.3J.1
00000dc0	00	00	00	00	00	4a	3e	aa	23	10	00	41	6C	65	78	00	J>*#. Alex.
00000dd0	00	36	00	08	00	04	00	ef	be	30	3e	0f	23	4a	3e	aa	.6ï%0>.#J>*
00000de0	23	2a	00	00	00	4f	00	00	00	00	00	03	00	00	00	00	#*0
00000df0	00	00	00	00	00	00	00	00	00	00	00	41	00	6C	00	65	Å.l.e
00000e00	00	78	00	00	00	14	00	7e	00	31	00	00	00	00	00	86	.x~.1t
00000e10	3e	b4	b5	11	00	50	69	63	74	75	72	65	73	00	00	66	>'µ. Pictures .f
00000e20	00	08	00	04	00	ef	be	30	3e	11	23	86	3e	b4	b5	2a	Ϊ%O>.#†>′μ*
00000e30	00	00	00	59	00	00	00	00	00	03	00	00	00	00	00	00	Ÿ
00000e40	00	00	00	3c	00	00	00	00	00	50	00	69	00	63	00	74	<p.i.c.t< td=""></p.i.c.t<>
00000e50	00	75	00	72	00	65	00	73	00	00	00	40	00	73	00	68	u.r.e.s0.s.h
00000e60	00	65	00	6c	00	6c	00	33	00	32	00	2e	00	64	00	6c	.e.1.1.3.2d.1
00000e70	00	6c	00	2c	00	2d	00	32	00	31	00	37	00	37	00	39	.1.,2.1.7.7.9
00000e80	00	00	00	18	00	56	00	31	00	00	00	00	00	8c	3e	3c	V.1@><
00000e90	b8	10	00	49	6e	74	65	72	бе	65	74	00	00	3e	00	08	Internet .>
00000ea0	00	04	00	ef	be	30	3e	ef	2b	8c	3e	3c	b8	2a	00	00	ï¾O>ï+Œ><,*
00000eb0	00	42	1c	01	00	00	00	01	00	00	00	00	00	00	00	00	.B
00000ec0	00	00	00	00	00	00	00	49	00	6e	00	74	00	65	00	72	I.n.t.e.r
00000ed0	00	6e	00	65	00	74	00	00	00	18	00	00	00	00	00	00	.n.e.t
00000ee0	25	00	00	00	18	00	00	00	00	1f	00	00	00	09	00	00	¥
00000ef0	00	49	00	бe	00	74	00	65	00	72	00	6e	00	65	00	74	I.n.t.e.r.n.e.t
00000100	00	00	00	00	00	25	00	00	00	0b	00	00	00	00	1f	00	

Fig. 9. Path fragments in the Jump List

Header
1100001
Padding
1 adding
Eile noth free and on to
File path fragments
E 11 C1 1
Full file paths
-
Padding
8
File path fragments
The pull hughlenes
Full file naths
i un me paulo
1

Table 4. Rough Jump List file structure

3 Conclusions and Recommendations

Jumps Lists have a number of practical applications for investigators. At minimum, they provide a list of files uploaded, downloaded, viewed, created, or otherwise used by every program on the system. Additionally, they can also serve as a log of frequent tasks undertaken by the user with some programs, such as with sites frequently visited on Internet Explorer. What's more, with further research it may be possible to determine how the file was processed through the program it is listed in, such as uploaded vs. downloaded, or in the case of a word processor, created vs. opened. In terms of practical application, this could mean proving an illegal image was uploaded to a server rather than downloaded, or that a ransom note was written and saved rather than merely opened. Any situation that needs to prove that a file was used on a system could potentially benefit from Jump List data.

Future research will ideally lead to the development of a tool that can automatically process Jump Lists as evidence. To reach this point, a number of research goals must first be accomplished. First, deciphering the 16 character identifier in the Jump List name will allow the identification and classification of Jump List files. Failing that, a comprehensive list of programs and their associated identifiers could be developed, however this is not ideal. Second, a greater understanding of how Jump Lists operate must be attained. The internal structure of the list must be deciphered to identify what information is actually stored by the list, other than the file name, host name, and path to the file. Finally, a program must be developed to automatically identify a list, extract the file paths, and tag each file with whatever attributes can be identified (uploaded, downloaded, etc). With the completion of all these tasks, Jump Lists will be of great use to investigators.

APPENDIX A – DETERMINING JUMP LIST ASSOCIATION

						×								
😋 🖉 📲 « AppData 🕨 Roaming 🕨 Microsoft 🕨 Windows 🕨 Recent Items 🕨 🔹 🤸														
Organize 🔻 Include in li	iibrary ▼ Share with ▼ Burn New folde	r				?								
☆ Favorites	Name	Date modified	Туре	Size										
📃 Desktop 😑	1b4dd67f29cb1962.automaticDestinations-ms	4/16/2011 4:14 PM	AUTOMATICDEST	12 KB										
🚺 Downloads	5c450709f7ae4396.automaticDestinations-ms	4/15/2011 1:48 AM	AUTOMATICDEST	15 KB										
Recent Places	9b9cdc69c1c24e2b.automaticDestinations-ms	4/14/2011 9:37 PM	AUTOMATICDEST	4 KB										
	a7bd71699cd38d1c.automaticDestinations-ms	4/16/2011 4:14 PM	AUTOMATICDEST	8 KB										
🥽 Libraries														
Documents														
J Music														
📔 Pictures 🗸														
4 items														

Fig. A.1. The AutomaticDestinations Folder

To determine which file belongs to which program, first view the selected program's Jump List in the Windows UI. Firefox was chosen for this example, as shown in figure A.2.



Fig. A.2. Firefox Jump List

From here, open each Jump List file in a HEX editor and search for the entries present in the list. By this method, we can determine which file belongs to which program.

🛛 C:\I	Jsers\A	lex\A	ppD	ata\	Roar	ning	\Mic	rosot	ft∖Wi	ndov	ws\R	ecen	t∖Au	toma	aticD	estin	atio	ns\5c450709f7ae4396.automaticDestinations-ms - Notepad++
File	Edit S	earc	h ۱	/iew	En	codi	ng	Lan	guag	e S	Settir	ngs	Ma	cro	Run	Т	extF)	X Plugins Window ?
	<u>-</u>			I,		00	- D	6	i e		i	ñ ¹	8	٢	3		-	i 🎫 🕦 🎼 😰 💿 🗉 🕨 🔤 🕿 🔺 🔻 🗷 Ħ 🗟 💖
😑 5c	450709	7ae4	396.a	autom	natic	Destin	nation	s-ms	1									
Add	ress	0	1	2	3	4	5	6	7	8	9	a	b	с	d	е	f	Dump
0000	01440	00	64	de	76	ac	10	00	00	00	00	43	3a	5c	55	73	65	.dÞvC:\Use
0000	01450	72	73	5c	00	00	22	00	00	00	02	00	00	00	14	00	00	rs\"
0000	01460	00	00	00	00	00	00	00	02	00	5c	5c	4d	41	47	55	53	\\ MAGUS
0000	01470	5c	55	73	65	72	73	00	41	6c	65	78	5c	50	69	63	74	\Users.Alex\Pict
0000	01480	75	72	65	73	5c	49	6e	74	65	72	6e	65	74	5c	63	61	ures\Internet\ca
0000	01490	74	2e	4a	50	47	00	39	00	00	00	09	00	00	a0	2d	00	t.JPG.9
0000	014a0	00	00	31	53	50	53	55	28	4c	9f	79	9f	39	4b	a8	d0	1SPSU(LŸyŸ9K"Đ
0000	014b0	e1	d4	2d	e1	d5	£3	11	00	00	00	07	00	00	00	00	Ob	áÔ-áŐó
0000	014c0	00	00	00	ff	ff	00	00	00	00	00	00	00	00	00	00	60	····ÿÿ······``
0000	014d0	00	00	00	03	00	00	aO	58	00	00	00	00	00	00	00	6d	Xm
0000	014e0	61	67	75	73	00	00	00	00	00	00	00	00	00	00	00	72	agusr
0000	014f0	16	82	a6	56	00	d1	4e	96	a6	ec	18	cc	4a	d2	59	b1	.,¦V.ÑN-¦ì.ÌJÒY±
0000	01500	e6	bf	a6	2f	21	e0	11	8f	b9	00	1a	a0	8a	a9	c5	72	濦/!à`Š©År
0000	01510	16	82	a6	56	00	d1	4e	96	a6	ec	18	cc	4a	d2	59	b1	.,¦V.ÑN-¦ì.ÌJÒY±
0000	01520	e6	bf	a6	2f	21	e0	11	8f	b9	00	1a	a0	8a	a9	c5	00	æ;¦/!๚©Å.
0000	01530	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000	01540	01	00	00	00	04	00	00	00	00	00	00	00	00	00	80	40	€0
0000	01550	04	00	00	00	00	00	00	00	04	00	00	00	00	00	00	00	· · · · · · · · · · · · · · · · · · ·
0000	01560	84	f2	99	ed	ae	e0	68	44	72	16	82	a6	56	00	d1	4e	"ò™í®àhDr.,¦V.ÑN
0000	01570	96	a6	ec	18	cc	4a	d2	59	7e	6b	94	d0	bb	66	e0	11	-¦ì.ÌJÒY~k″Đ»fà.
0000	1580	99	e6	e3	3a	06	b7	be	d2	72	16	82	a6	56	00	d1	4e	™æã:. %Òr.,¦V.ŇN
0000	01590	96	a6	ec	18	cc	4a	d2	59	7e	6b	94	dO	bb	66	еO	11	-¦ì.ÌJÒY~k″Đ»fà.
0000	015a0	99	e6	e3	Зa	06	b7	be	d2	6d	61	67	75	73	00	00	00	™æã:. %Òmagus
0000	015b0	00	00	00	00	00	00	00	00	04	00	00	00	00	00	00	00	· · · · · · · · · · · · · · · · · · ·
0000	015c0	00	00	80	3f	84	11	cf	ac	30	fb	$^{\rm cb}$	01	ff	ff	ff	ff	€?".Ï⊣OûË.ÿÿÿÿ
0000	015d0	26	00	43	00	3a	00	5c	00	55	00	73	00	65	00	72	00	€.C.:.\.U.s.e.r.
0000	015e0	73	00	5c	00	41	00	6c	00	65	00	78	00	5c	00	44	00	s.\.A.l.e.x.\.D.
0000	015f0	6f	00	77	00	6e	00	6c	00	6f	00	61	00	64	00	73	00	o.w.n.l.o.a.d.s.
0000	01600	5c	00	48	00	78	00	44	00	53	00	65	00	74	00	75	00	\.H.x.D.S.e.t.u.
0000	01610	70	00	45	00	4e	00	2e	00	7a	00	69	00	70	00	cf	c2	p.E.Nz.i.p.ÏÂ
0000	01620	5a	54	35	b3	b3	Od	72	16	82	a6	56	00	d1	4e	96	a6	215 ⁻¹ .r., V.NN-;
0000	01630	ec	18	cc	4a	d2	59	7b	6b	94	d0	bb	66	e0	11	99	e6	ì.ÌJÒY{k″Đ»fà.™æ
0000	01640	4c	00	00	00	01	14	02	00	00	00	00	00	сO	00	00	00	LÀ
0000	01650	00	00	00	46	83	00	00	00	20	00	00	00	e3	88	5d	99	Ffä^]™
0000	01660	0d	fb	$^{\rm cb}$	01	e6	7b	6f	98	0d	fb	$^{\rm cb}$	01	c9	83	b4	99	.ûĔ.æ{oûĔ.Éf′™

Fig. A.3. File entries in the Firefox Jump List