

** The following information is a guide to understanding the Prefetch Folder and Windows Shortcut File Format (LNK) and all work undertaken in my research, should be verified by the analyst. Prior to release this product was tested in the Forensic Computing community, and some of their findings are incorporated into this guidance. All output from WFA should be verified by the analyst. **

Allan S Hay

November 2005

The program is divided into 3 parts, which the latter 2 I go into some depth to explain the workings of how the data is derived.

Thumbnail Analyser

This program in most circumstances, will extract the OLE embedded data from a Thumbs.db file and present the information in a visible format. The larger the size of file being examined, directly affects the time for the Objects to be rendered.

Although it may look that the file is not being processed, be patient. Select a single image (Save) to be exported, or select (Report). The number on the top left of the report (Count) is the number of images rendered.

For a thorough explanation of the Forensic value of the Thumbs.db structure an excellent source of information can be found at:

http://www.accessdata.com/files/whitepapers/tdb.pdf

Prefetch Analyser

The Prefetch cache is resident on a Windows XP OS. Microsoft created a Prefetch cache to improve boot and application launch time. The functionality of how the cache performs is based on a Registry key which can be found here (HKEY_LOCAL_MACHINE\ SYSTEM\ CurrentControlSet\ Control\ Session Manager\Memory Management\PrefetchParameter).

The Prefetch cache purpose is to speed up processes at boot and during application run time. By cachingcommonly used applications the OS can determine to apportion system resources in anticipation that the user will access the application. When an application is launched the system updates an entry in the path **C:/Windows/Prefetch** with the name of the application and a file extension (.pf). The file contains among other items the last time that the file was modified as a 64bit HEX value time, and increments an integer on how many times the application has been run.

Before I explain where the information is, please bear in mind that the Prefetch cache is taking information from the boot process and can take information from Scheduled Tasks. Therefore, it would be prudent to access the RUN (NTUSER/Software/Microsoft/Windows/Explorer/RUN) key in the registry and also the TASKS (C:/Windows/Tasks). After this scour the drive to determine if there is any 3rd party software scheduling resident on the drive.

It would appear that in system idle time every 3-5 days (but can be longer), depending on how the caching is determined, the folder will purge itself. The analyst can search across the drive using a GREP expression and pull out the data from the Unallocated Clusters. By scrutinising the data structure the analyst should be able to determine the data from the two forgoing offsets, minus temporal data.

The following should be used to search across the Unallocated Clusters to determine the historic Prefetch Files.

(....SCCA.)

or:

("\\x11\\x00\\x00\\x00\\x53\\x43\\x41\\x0F", FileClass::GREP);

Using a hex editor program, open a file in the Prefetch folder, try finding an .exe that you would use regularly, perhaps MS Word. In your hex editor, go to File Offset 120. You will see an 64 bit hex value, which is the date that the executable was last summoned, and this gets updated in the PF file. Go to File Offset 144, note the value. This is the number of times that the application has been run, since the creation of the .pf file.

This is an OS dependent folder, so the examiner has to understand that PFA will only work on Windows XP. Hopefully Windows Vista will include a Prefetch or equivalent. Before using PFA, it is best to see where the data you are interpreting is located, and using a hex editor you will be able to see the changes being made. MiTec HexEdit can be downloaded from the link below:

http://www.mitec.cz/Downloads/HEXEdit.zip

The research work that I have carried out has been evolved into Version 1.0 of the Prefetch Folder Analyser, which Michal Mutl has coded for me. The program will automatically read the local drive of the user when first initialized. Within the main viewing pane is the breakdown of the resident data. Navigate with Hexedit to your Prefetch Folder on your OS and select a file which is in regular use. Open a .pf file in HexEdit and you should see an image similar to the one on the following page.

Figure 1.0

The following is an image captured from the Winword.pf when viewed via Hexedit. Note the File Offset 120 which is a 64 bit Windows Time entry. By placing the cursor on the first byte, the Hexedit program enumerates the string and this can be seen on the left hand panel. Note this time for later.

Figure 1.0

The following is an image captured from the Winword.pf when viewed via Hexedit. Note the File Offset 120 which is a 64 bit Windows Time entry. By placing the cursor on the first byte, the Hexedit program enumerates the string and this can be seen on the left hand panel. Note this time for later.

🖥 MiTeC Hexa	decimal Editor - [WINWORD.EX	E-37F6AE09.pf]	
File Edit Fo	rmat Tools Windows Help		- 8 ×
St WINWOR	D.EXE-37F6AE09.pf		
- Position		0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 01234567 9ABCDEF	A
+ Start Offset	0	0x00000 1100 0000 5343 4341 0F00 0000 72C3 01003CCArÃ	
+ Position	120	0x00010 5700 4900 4E00 5700 4F00 5200 4400 2E00 W.I.N.W.O.R.D	
+ Selection	0	0x00020 4500 5800 4500 0000 0000 0000 0000 E.X.E	
- Data		0x00030 3C00 3E00 9C28 5C80 4B00 4C00 08FD 86E1 <.>.@(\€K.L\†á	
Signed 8	102	0x00040 E800 F577 0300 0000 0000 F577 09AE F637 2.5w	
E Unsigned 8	102	0×00050 0000 0000 9800 0000 7E00 0000 700A 0000"	
E Signed 16	-418	0×00060 B91D 0000 106F 0100 6245 0000 8084 0100 *	
Unsigned 16	65126	0×00070 0100 0000 F205 0000 H6FE 0392 0FC5 C501 0. Bb. (AA.)	
E Signed 32	-1645232026		
E Unsigned 32	2449735270		
Eligned 64	127724835114843750		
Float 32	0.00		
Fillar 64	0.00		
- Timestamps	20/20/2006 18:05:11		
LIBIN 22	23/03/2005 10:03:11		
DOS 22	29/09/2006 4 8:05:14		
Summary	23/03/2003 10:03.11	0x000F0 A401 0000 3500 0000 0400 0000 A700 0000 #SS	
CRC 32	DB744A94	0x00100 2600 0000 1002 0000 3500 0000 0200 0000 4	
MDS	096A43D271940D41AE93A173	0x00110 CD00 0000 2500 0000 7C02 0000 3300 0000 I« 3	
Data Size	115570 B	0x00120 0200 0000 F300 0000 1500 0000 E402 0000oa	
Yester Content		0x00130 3200 0000 0200 0000 0801 0000 2900 0000 2))	
		0x00140 4A03 0000 3300 0000 0200 0000 3101 0000 J31	
		0x00150 5300 0000 B203 0000 3200 0000 0200 0000 S*2	
		0x00160 8401 0000 2F00 0000 1804 0000 3300 0000/3	
		0×00170 0200 0000 B301 0000 0200 0000 8004 0000³€	
		0x00180 3200 0000 0400 0000 B501 0000 D80B 0000 2µØ	
		0x00190 E604 0000 5400 0000 0600 0000 8D0D 0000 æTD	
		0x001A0 3700 0000 9005 0000 4F00 0000 0400 0000 7D0	
		0x001B0 C40D 0000 1400 D000 3006 0000 3400 0000 Å04	
		0x001C0 0600 0000 D80D 0000 3D00 0000 9A06 0000Ø=š	
		0x00100 1000 0000 0500 0000 150E 0000 2800 0000	
		0x001E0 D406 0000 8100 0000 0100 0000 3D0E 0000 0D	
		0x001F0 1000 0000 0807 0000 3400 0000 0200 0000Ø4	
NERSE PROVING		0x00200 5A0E 0000 0100 0000 4208 0000 3500 0000 ZB	
Swap Endian	Mark Zeros	0×00210 0400 0000 580E 0000 1800 0000 AE08 0000	
		0x00220 4800 0000 0600 0000 730E 0000 1F00 0000 H	
	6	0×00230 4009 0000 3400 0000 0200 0000 920E 0000 84	
	24	0×00240 0200 0000 4409 0000 3400 0000 0400 0000	
	Second -	0×00250 9405 0000 5500 0000 1404 0000 5000 0000 * * 1	
	PFFIe		
			~

This image is the same as the previous, but File Offset 144 is highlighted which is the incremental integer. The value interpreted with Hexedit can be seen in the left hand panel in the Data section.

🖇 MiTeC Hexadecimal	_ 7 🗙		
File Edit Format To	ols Windows Help		283
NINWORD.EXE-3	7F6AE09.pf		
Position		0001 0203 0405 0607 0809 0408 0C00 0F0F 12345678948C0FF	
+ Start Offset 0			
+ Position 144			
E Selection 0		0×00010 3700 4500 0000 0000 0000 0000 0000 FTW.W.G.K.D	
🗄 Data 🔄			
E Signed 8 8			
E Unsigned 8 8			
E Signed 16 8		0x000s0 0000 0000 9800 0000 /200 0000 /004 0000p	
E Unsigned 16 8		0x00060 B91D 0000 106F 0100 6245 0000 80B4 01008BEe	
E Signed 32 8	T	0x00070 0100 0000 F20E 0000 66FE 0392 0FCS CS01oFpAA.	
Unsigned 32 8		0884080 0000 0000 0000 0000 0000 0000 0	
E Signed 64 171798	69192	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	
Float 32 0.00		0x000A0 0000 0000 3200 0000 0200 0000 3400 000024	
Float 64 0.00		0x000B0 4100 0000 6600 0000 3500 0000 0200 0000 Af5	
Timestamps	In the second	0x000C0 7500 0000 0500 0000 D200 0000 3400 0000 u	
VMIN 64 02/01/11	601 23:31:22	0x00000 0400 0000 7A00 0000 2900 0000 3C01 0000z)<	
UNIX 32 01/01/1	970 00:00:08	0x000E0 3300 0000 0400 0000 A300 0000 0400 0000 3ź	
DOS 32 02/01/11	601 23:31:22	0x000F0 Å401 0000 3500 0000 0400 0000 Å700 0000 ×5\$	
- Summary		0x00100 2600 0000 1002 0000 3500 0000 0200 0000 «5	
CRC 32 DB744A	194	0x00110 CD00 0000 2600 D000 7C02 0000 3300 0000 1«	
MD5 096A43	D271940D41AF93A173	0x00120 0200 0000 F300 0000 1500 0000 E402 0000	
Data Size 115570	B	0x00130 3200 0000 0200 0000 0801 0000 2900 0000 2	
		0x00140 4403 0000 3300 0000 0200 0000 3101 0000 J31	
		0x00150 5300 0000 8203 0000 3200 0000 0200 0000 8 * 2.	
		0x00150 8401 0000 2500 0000 1804 0000 3300 0000 / 3	
		0x001E0 C40D 0000 1400 0000 3006 0000 3400 0000 A04	
		0x001C0 0600 0000 0800 0000 3000 0000 9406 00009=s	
		0x001D0 1C00 0000 0500 0000 150E 0000 2800 0000(
		0x001E0 D406 0000 8100 0000 0100 0000 3D0E 0000 0D	
		0x001F0 1D00 0000 D807 0000 3400 0000 0200 0000	
C Swan Endian	Wark Zaros	0x00200 SADE 0000 0100 0000 4208 0000 3500 0000 ZBS	
1 Swap Linuari	Wark Zerus	0x00210 0400 0000 5B0E 0000 1800 0000 AE08 0000[
		0x00220 4800 0000 0600 0000 730E 0000 1F00 0000 Hs	
-	1	0x00230 4009 0000 3400 0000 0200 0000 920E 0000 04	
		0x00240 0200 0000 A&09 0000 3400 0000 0400 0000*4	
PF F	ie	UX00250 940E 0000 5E00 0000 140A 0000 5D00 0000 "^	
	Matt	0x00260 0400 0000 F20E 0000 0100 0000 D00A 0000bb	
		0x00270 5F00 0000 0400 0000 F30E 0000 0100 0000	
		0x00270 5F00 0000 0400 0000 F30E 0000 0100 0000 =	N

As can be seen from the header information I have accessed a .pf file for WinWord (MS Word). I then run MS Word, only the application is needed, not a proprietary file type associated with MS Word. I do not have to open a document, only the application needs to be summoned.

Using Hexedit it is possible to differentiate between 2 files, when placed within Hexedit for comparison. I loaded the original Winword.pf and a previously saved Winword .pf file prior to using MS Word. It is possible to see the data that has changed. In this case look at the two 64 bit File Time differences. Both are highlighted in the panel on the left.

그 🖙 님 🛢		×		
8 Winword	1.pf 🔡 WINWORD.EXE-3	7F6AE0	9.pf	
signed 32	+1043232020	A [0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 01234567 9ABCDEF	1
Unsigned 32	2449735270	17.201		
Signed 64	127724835114843750		1100 0000 5543 4341 0F00 0000 7213 0F00SULAFA	
Float 32	0.00		0x00010 5700 4900 4E00 5700 4F00 5200 4400 2E00 W.I.N.W.O.R.D	
Float 64	0.00		0x00220 4500 5800 4500 0000 0000 0000 0000 E.X.E	
Timestamps			UX00030 3C00 3E00 9C28 5C80 4B00 4C00 08FD 86E1 <.>.@(\€K.Lyta	
VMN 64	29/09/2005 16:05:11		0x00040 E800 F577 0300 0000 0000 F577 09AE F637 e.ow	
UNIX 32	18/08/2047 10:01:10		0x00050 0000 0000 9800 0000 7E00 0000 700A 0000	
DOS 32	29/09/2005 16:05:11		0x00060 B91D 0000 1C6F 0100 6245 0000 80B4 0100 ⁴ obE€	
Summary	1007111001	-	0100 0000 F20E 0000 66FE 0392 0FC5 C5016fpAA.	
SRC 32	DB744A94		0x00080 0000 0000 0000 0000 0000 0000 0	
Swap Endian	Mark Zeros		0x00090 0800 0000 0400 0000 0000 0000 3400 00004	
			0x000A0 0000 0000 3200 0000 0200 0000 3400 000024	
			0x000B0 4100 0000 6600 0000 3500 0000 0200 0000 Af5	
	124		0x000C0 7500 0000 0500 0000 D200 0000 3400 0000 ud4	
	to the second se		0x000D0 0400 0000 7A00 0000 2900 0000 3C01 0000z)<	
	PF File		0x000E0 3300 0000 0400 0000 A300 0000 0400 0000 3ź	
			0x000F0 A401 0000 3500 0000 0400 0000 A700 0000 ×5\$	
			0x00100 2600 0000 1002 0000 3500 0000 0200 0000 «5	
C:\Docur	ments and Settings\ASH\Desktop	Winword:	Lof	
	and the second	A		
WINWORD.	EXE-37F6AE09.pf			
WINWORD.	EXE-37F6AE09.pf	~	0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 01234567 9ABCDEF	
WINWORD. onsigned 52 Signed 64	EXE-37F6AE09.pf 1144303052 127724846695000000 764.52	-	0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100SCCAÅ	
WINWORD. Onsigned 32 Signed 64 Float 32 Elect 54	EXE-37F6AE09.pf 1194303032 127724846695000000 764.62	1	0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100SCCAÅ 0x00010 5700 4900 4E00 5700 4F00 5200 4400 2E00 W.I.N.W.D.R.D	
WINWORD. Unsigned 32 Signed 64 Float 32 Float 64 Timestempe	EXE-37F6AE09.pf 1144909652 127724846695000000 764.62 0.00		0001 0203 0405 0607 0809 0A08 0C0D 0E0F 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100sCCAÅ 0x00010 5700 4900 4E00 5700 4F00 5200 4400 2E00 W.T.N.W.O.R.D 0x00020 4500 5800 4500 0000 0000 0000 0000 V.K.E	
WINWORD. onsigned 32 Signed 64 Float 32 Float 64 Timestamps	EXE-37F6AE09.pf 1144903052 12772484669500000 764.62 0.00 200000000545.24.20		0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100SCCAÅ 0x00010 5700 4900 4E00 5700 4F00 5200 4400 2E00 W.T.N.W.O.R.P.L 0x00020 4500 5800 4500 0000 0000 0000 0000 0E.X.E	
WINWORD. Cinsigned 32 Signed 64 Float 32 Float 64 Timestamps VMN 64 UNIX 22	EXE-37F 6AE09.pf 1144939532 127724846695000000 764.62 0.00 29/09/2005 16:24-29 440/0005 06:40.32		0001 0203 0405 0607 0809 0A0B 0COD 0E0F 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAA. 0x00010 5700 4900 4500 5700 4400 2200 W.I.N.W.O.R.D 0x00020 4500 5800 4500 0000 0000 0000 E.X.E. 0x00030 3600 9228 5280 4800 470 5821 <	
WINWORD. Cinsigned 32 Signed 64 Float 32 Float 64 Timestamps VMN 64 UNIX 32 DOS 23	EXE-37F 6AE09. pf 1144935552 12772446655000000 764.62 0.00 29/09/2005 16:24:29 14/04/2006 04:40:32 29/09/2005 46:32:20		0001 0203 0405 0607 0809 0A0B 0COD DEOF 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAÅ A 0x00010 5700 4900 4500 5700 4900 0000 000 W.I.N.W.O.R.D 0x00020 4500 5800 4500 0000 0000 0000 EX.E 0x00030 3C00 3E00 9228 5680 4B00 4200 8E1 <.>.e(\€K.Lýtá 0x00040 E800 F577 0300 0000 0007 F57 94.0 0007	
WINWORD. Unsigned 32 Signed 64 Float 32 Float 64 Timestemps VNN 64 UNIX 32 DOS 32 Supposed	EXE-3756AE09.pf 1194938532 12772446659500000 764.62 0.00 29409200516:24-29 14/04/200516:24-29 14/04/200516:24-29		0001 0203 0405 0607 0809 0A0B 0COD 0E0F 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAA 0x00010 5700 4900 4500 5200 4400 2E00 W.T.N.W.O.R.P.L 0x00020 4500 3800 4500 0000 0000 000 EX.E 0x00030 3600 928 5600 4500 086E1 <	
WINWORD. Crisigned 52 Signed 64 Float 32 Float 64 Timestamps VIN 64 UNIX 32 DOS 32 Summary CPC 32	EXE-37F 6AE09, pf 11449395032 12772446655000000 764.62 0.00 29.09/2005 16:24:29 14.04/2006 04:40.32 29.09/2005 16:24:29 EC3DE 409		0001 0203 0405 0607 0809 0A0B 0COD 0E07 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAA A 0x00010 5700 4900 4200 5700 4400 2200 W.I.N.W.O.R.D 0x00020 4500 5800 4500 0000 0000 0000 E.X.E 0x00030 5000 5200 400 250 E.X.E Yi 0x00040 E800 577 0300 0000 0400 6737 è.Gu A Yi 0x00050 0000 570 0900 577 0300 600 577 Yi A A A A A A A A A A A A A A A A A A A A A A <td></td>	
WINWORD. Unsigned 32 Signed 84 Float 32 Float 64 Timestamps VNN 64 UNIX 32 DOS 32 Summary CRC 32 MDS	EXE-37F 6AE09.pf 1149838032 1277244665900000 764.62 0.00 29/03/2005 16:24:29 14/04/2006 04:40:32 29/09/2005 16:24:29 FC3DEA98 FC3DEA98		0001 0203 0405 0607 0809 0A0B 0COD 0EOF 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAÅ A 0x00010 5700 4900 4200 5200 4400 2200 W.T.N.W.O.R.D 0x00030 3C00 5800 4500 0000 0000 621 5 C 0x00030 3C00 5228 5268 4B00 4C00 0870 8621 <	
WINWORD. Crisigned 64 Float 64 Float 64 Timestamps VNN 64 UNIX 32 DOS 32 Summary CRC 32 MD5.	EXE-37F 6AE09, pf 11449398032 12772446655000000 764.62 0.00 2909200516:24-29 1404/200604:40:32 2909/200516:24-29 FC3DEA98 E07DBE1270CCF5B436890		0001 0203 0405 0607 0809 0A0B 0COD 0EOF 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 \$CCAA 0x00010 5700 4900 4500 5700 4700 5200 400 2E0 U.T.N.W.O.R.P.D 0x00020 4500 800 4500 0000 0000 000 EX.E. 0x00030 3C00 928 5260 400 2E06 V.T.N.W.O.R.P.L 0x00030 3C00 928 5260 400 086E1 <	
WINWORD. Crisigned 52 Signed 64 Float 32 Float 32 Float 64 Timestamps VNN 64 UNIX 32 DOS 32 Strumary CRC 32 MD5 Swap Endiar	EXE-37F 6AE09, pf 11449395032 12772446655000000 764.62 000 29/09/2005 16:24:29 14/04/2006 04:40.32 29/09/2005 16:24:29 FC3DEA98 E07DE1270CCF5B436690 Mark Zeros		0001 0203 0405 0607 0809 0A0B 0COD 0E07 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAA A 0x00001 5700 4900 4200 5700 4700 5200 4400 2E00 W.I.N.W.O.R.D 0x00020 4500 5800 4500 0000 0000 0000 EX.E.	
WINWORD. onsigned 32 Signed 64 Float 32 Float 64 Timestamps WIN 64 UNIX 32 DOS 32 Summary CRC 32 MD5 Swap Endiar	EXE-37F 6AE09, pf 1149836532 12772446655000000 764.62 0.00 29/03/2005 16:24:29 14/04/2006 04:40:32 29/09/2005 16:24:29 FC3DEA98 E07DBE1270CCF5B43669D Mark Zeros		0001 0203 0405 0607 0809 0A0B 0C00 0E0F 01234567 9ABCDEF 0x00001 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAÅ 0x00001 5700 4900 4200 5700 4900 2000 0000 0000 0000 0000 500	
WINWORD. Orisigned 32 Signed 64 Float 32 Float 54 Timestamps VMN 64 UNIX 32 DOS 32 Summary CRC 32 MD5 Swap Endiar	EXE-37F 6AE09. pf 11449398032 12772446655000000 764.62 0.00 29/09/2005 16:24-29 14/04/2006 04:40:32 29/09/2005 16:24-29 FC3DEA98 E07DBE1270CCF5B436680 Mark Zeros		0001 0203 0405 0607 0809 0A0B 0COD 0EOF 01234567 9ABCDEF 0x00001 1100 0000 5343 4341 0F00 0000 C4B7 0100 \$CCAA A 0x00010 5700 4900 4200 5700 4700 5200 4400 2200 U.T.N.W.O.R.P.D 0x00020 4500 3800 4500 0000 0000 0000 EX.E. 0x00030 3C00 3200 0000 7700 0342 6521 <	
WINWORD: unsigned 32 Signed 64 Float 32 Float 64 Timestamps vNN 64 UNIX 32 DOS 32 Summary CRC 32 MD5 Swap Endiar	EXE-37F 6AE09, pf 1449395032 12772446655000000 764.62 0.00 29092005 16:24:29 14:04/2006 04:40.32 29092005 16:24:29 FC3DEA98 E07DEE1270CCF5B43669D Mark Zeros		0001 0203 0405 0607 0809 0A0B 0C00 DEOF 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100	
WINWORD. Unsigned 52 Signed 54 Float 32 Float 54 UNIX 32 DOS 32 Summary CRC 32 MD5 Swap Endiar	EXE-3756AE09.pf 1494938032 12772446685000000 764.62 0.00 29409200516:24-29 1404/200504:40:32 29.09/200516:24-29 FC3DEA98 E07/DEFL27/0CCF5B43669D Mark Zeros		0001 0203 0405 0607 0809 0A0B 0COD 0EOF 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAA 0x00001 5700 4900 4200 5700 4F00 5200 4400 2E00 V.T.N.W.O.R.D 0x00020 4500 5800 4500 0000 0000 EX.E 0x00030 6800 9577 0300 0000 770 0542 F537 à.bw Bw.e67 0x00050 0000 0000 6840 0000 I.e Bw.e67 0x00050 0000 0000 6000 0000 I.e I.e I.e 0x00050 0000 0000 0000 0000 0000 I.e I.e I.e I.e 0x00050 0000 0000 0000 0000	
WINWORD. Unsgreet 32 Signed 84 Float 32 Float 64 UNIX 64 UNIX 32 DIX 32 Summary CRC 32 MD5 Swap Endiar	EXE-37F 6AE09, pf 11449398032 12772446685500000 764 62 000 29092005 16:24-29 1404/2006 04:40:32 2909/2005 16:24-29 FCSDEA98 E07DBE1270CCF5B436690 Mark Zeros PF File		0001 0203 0405 0607 0809 0A0B 0COD 0EOF 01234567 9ABCDEF 0x00001 1100 0000 5343 4341 0F00 0000 C4B7 0100 \$CCAA. A 0x00010 5700 4900 4200 5700 4700 5200 4400 2200 U.T.N.W.O.R.A. A 0x00020 4500 8000 4500 0000 0000 0000 000 EX.E. A 0x00020 4500 5000 0000 0000 86E1 <	
WINWORD. Unsgreed 32 Signed 64 Float 32 Float 64 UNIX 32 JOS 52 Summary CRC 32 Swap Endlar	EXE-37F 6AE09, pf 1449395032 12772446655000000 764.62 0.00 29092005 16:24:29 14/04/2006 04:40.32 29092005 16:24:29 FC3DEA98 E07DBE1270CCF5B43665D TMark Zeros PF File		0001 0203 0405 0607 0809 0A0B 0COD DEOF 01234567 9ABCDEF 0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100 SCCAA A 0x00010 5700 4900 4200 5700 4900 200 0000 0000 0000 0000 EX.E. A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A	

Figure 1.2

For your information when using MiTec HexEdit. When the user compares two files, the area shaded in green represents the comparative differences between the two files.

The last image in the section shows the differences as at File Offset 144. In Figure 1.0, the integer was showing a value of 8 runs. Note that I used MS Word on only one occasion and the value has incremented by one.

🚔 MiTeC Hexadecimal Editor		- 7 🛛
File Edit Format Tools Windows Help		14
	<u>k</u>	
Winword1.pf WINWORD.EXE-37F6A	JE09.pf	
Position	0001 0203 0405 0607 0800 0808 0000 0605 123456780880055	~
+ Start Offset 0		
+ Postion 144	0x00000 1100 0000 5343 4341 0700 0000 7243 0100	12 A
+ Selection 0	0x00010 3700 4900 4200 3700 4200 3200 4400 2200	
- Data	0x00020 4300 3000 4300 4000 0000 0000 0000	
+ Signed 8 8		
+ Unsigned 8 8		
+ Signed 16 8	D×00050 B91D 0000 105F 0100 6245 0000 8084 0100 *o. bF€'	
+ Unsigned 15 8	0x00070 0100 0000 F20E 0000 66FE 0392 0FC5 C501 b. fb. f.Å.	
+ Signed 32 8	0x00080 0000 0000 0000 0000 0000 0000 0	
Swap Endian Mark Zeros	0800 0000 0400 0000 0000 0000 3400 00004	
	0x000A0 0000 0200 0000 0200 0000 3400 0000	
	0x00080 4100 0000 6600 0000 3500 0000 0200 0000 Af5	
₽	0x000C0 7500 0000 0500 0000 D200 0000 3400 0000 u	
	0x00000 0400 0000 7A00 0000 2900 0000 3C01 0000z)<	
PF File	0x000E0 3300 0000 0400 0000 A300 0000 0400 0000 3	
	0x000F0 A401 0000 3500 0000 0400 0000 A700 0000 ×5\$	
	0x00100 2600 0000 1002 0000 3500 0000 0200 0000 65	~
C:\Documents and Settings\ASH\Deskton\\\\\insur	ord) of	
	лаар	
👪 WINWORD.EXE-37F6AE09.pf		
E Position	0001 0203 0405 0607 0809 0A0B 0C0D 0E0F 123456789ABCDEF	
Start Offset	0x00000 1100 0000 5343 4341 0F00 0000 C4B7 0100SCCAA	1
Position 144	0x00010 5700 4900 4E00 5700 4F00 5200 4400 2E00 W.I.N.W.O.R.D	
E Selection 0	0x00020 4500 5800 4500 0000 0000 0000 0000 000	
E Data	0x00030 3C00 3E00 9C28 5C80 4B00 4C00 08FD 86EJL <.>.∞(\€K.Lý†á	
+ Signed 8 9	0x00040 E800 F577 0300 0000 0000 F577 09AE F637 è.õwõw.eö7	
L Unsigned 8 9	Dx00050 0000 0000 9800 0000 7F00 0000 840A 0000	
Uppigped 16 9	0x00060 B91C 0000 3063 0100 C845 0000 F8A8 0100 4DcEE	
E Disigned 10 3	0x00070 0100 0000 CC0E 0000 C027 3F44 12C5 C501 1 A 2D. AA.	
T Signed 32 3	9x00880 0000 0000 0000 0000 0000 0000 00	
Swap Endian Mark Zeros	0x00000 000 0500 0000 0000 3400 000044	
	0x000A0 0000 0200 0200 0200 0200 0400 000024	
	0x000B0 4100 0000 6600 0000 3500 0000 0200 0000 Af5	
₽	0x00000 7500 0000 0500 0000 0200 0000 3400 0000 u	
	0x00000 0400 0000 7A00 0000 2900 0000 3C01 0000z)<	
PF File	0x000E0 3300 0000 0400 0000 A300 0000 0400 0000 3£	
	0x000F0 A401 0000 3500 0000 0400 0000 A700 0000 ×5	
	0x00100 2600 0000 1002 0000 3500 0000 0200 0000 &5	
C:\WINDOWS\Prefetch\WINWORD.EXE-37F6A	4E09.pF	
6		3
		1

<u>Using PFA</u>

The program has been developed to output the following information which may be of interest of the analyst.

Application:

This column highlights the name of the application or program. In research work, I found that some programs that are not installed, that is stand alone, will not have any entries in the Prefetch Folder. Or entries that are standalone can have erroneous runs.

Created Date:

When the system flushes the cache as and when it sees fit, (MS state 3 days, though I do not agree with this), a new .pf file should be generated when the program is either executed, or on next boot up, a new .pf file will created. The first time that the program is used the .pf will increment by one and so on. If the .pf file is not flushed, then from time the .pf file being created, along with the value at File Offset 144, should give an indication of how many times that the program has been used.

Written:

Easily understood as the time and date at which the file was written to.

Last Accessed:

This is not to be interpreted as the last time that the file was accessed to update, this could have been accessed by a third party utility.

Embedded Date:

This date as at File Offset 120, should marry with Written Date. The reason that the Embedded Date was extracted is due to the fact that when these files become purged, it is possible to search in the UC for these files given the header information. I have run a bastardized EnScript to pull out the information and I will make this available ASAP.

Runs:

The value/s returned at File Offset 144.

File Path Hash:

See Appended.

MD5:

A hash value calculated on the unique data embedded with each .pf file.

Practical Use.

When 'Analyse Prefetch' is selected it will automatically check for the default location of the Prefetch folder, normally C:/Windows/Prefetch.

The user should now see the entries of their local Prefetch Folder. Expand the Window to view all the detail.

To view a foreign Prefetch Folder, the volume must be mounted in either VM or proprietary product like Mount Image Pro, in order to preserve the MAC times.

Appendix

The information below was drafted from the MSDN site and was accurate at the date of publishing.

http://msdn.microsoft.com/msdnmag/issues/01/12/XPKernel/default.aspx

All versions of Windows except real-mode Windows 3x are demand-paged operating systems, where file data and code is faulted into memory from disk as an application attempts to access it. Data and code is faulted in page-granular chunks where a page's size is dictated by the CPU's memory management hardware. A page is 4KB on the *x*86. Prefetching is the process of bringing data and code pages into memory from disk before it's demanded.

In order to know what it should Prefetch, the Windows XP Cache Manager monitors the page faults, both those that require that data be read from disk (hard faults) and those that simply require that data already in memory be added to a process's working set (soft faults), that occur during the boot process and application startup. By default it traces through the first two minutes of the boot process, 60 seconds following the time when all Win32 services have finished initializing, or 30 seconds following the start of the user's shell (typically Microsoft Internet Explorer), whichever of these three events occurs first. The Cache Manager also monitors the first 10 seconds of application startup. After collecting a trace that's organized into faults taken on the NTFS Master File Table (MFT) metadata file (if the application accesses files or directories on NTFS volumes), the files referenced, and the directories referenced, it notifies the Prefetch component of the Task Scheduler by signalling a named event object.

In order to know what it should Prefetch, the Windows XP Cache Manager monitors the page faults, both those that require that data be read from disk (hard faults) and those that simply require that data already in memory be added to a process's working set (soft faults), that occur during the boot process and application startup. By default it traces through the first two minutes of the boot process, 60 seconds following the time when all Win32 services have finished initializing, or 30 seconds following the start of the user's shell (typically Microsoft Internet Explorer), whichever of these three events occurs first. The Cache Manager also monitors the first 10 seconds of application startup. After collecting a trace that's organized into faults taken on the NTFS Master File Table (MFT) metadata file (if the application accesses files or directories on NTFS volumes), the files referenced, and the directories referenced, it notifies the Prefetch component of the Task Scheduler by signaling a named event object.

The Task Scheduler then performs a call to the internal NtQuerySystemInformation system call requesting the trace data. After performing post-processing on the trace data, the Task Scheduler writes it out to a file in the \Windows\Prefetch folder. The file's name is the name of the application to which the trace applies followed by a dash and the hexadecimal representation of a hash of the file's path. The file has a .pf extension, so an example would be NOTEPAD.EXE-AF43252301.PF. An exception to the file name rule is the file that stores the boot's trace, which is always named NTOSBOOTB00DFAAD. PF (a convolution of the hexadecimal-compatible word BAADF00D, which programmers often use to represent uninitialized data).

Only after the Cache Manager has finished the boot trace (the time of which was defined earlier) does it collect page fault information for specific applications.

When the system boots or an application starts, the Cache Manager is called to give it an opportunity to perform Prefetching. The Cache Manager looks in the Prefetch directory to see if a trace file exists for the Prefetch scenario in question. If it does, the Cache Manager calls NTFS to Prefetch any MFT metadata file references, reads in the contents of each of the directories referenced, and finally opens each file referenced. It then calls the Memory Manager to read in any data and code specified in the trace that's not already in memory. The Memory Manager initiates all of the reads asynchronously and then waits for them to complete before letting an application's startup continue.

How does this scheme provide a performance benefit? The answer lies in the fact that during typical system boot or application startup, the order of faults is such that some pages are brought in from one part of a file, then from another part of the same file, then pages are read from a different file, then perhaps from a directory, and so on. This jumping around results in moving the heads around on the disk. Microsoft has learned through analysis that this slows boot and application startup times. By Prefetching data from a file or directory all at once before accessing another one, this scattered seeking for data on the disk is greatly reduced or eliminated, thus improving the overall time for system and application startup.

To minimize seeking even further, every three days or so, during system idle periods, the Task Scheduler organizes a list of files and directories in the order that they are referenced during a boot or application start, and stores the list in a file named \Windows\Prefech\Layout.ini.

LNK Shortcut Analyser

There are many references on the web to the file format of an LNK, and I have found the most informative to be the White Paper authored by Jesse Hager. The PDF can be downloaded at

http://www.i2s-lab.com/Papers/The_Windows_Shortcut_File_Format.pdf

The reason I carried out research into this area was due attempting to determine whether or not a user had accessed a CD which had CP images on it. I was able to see using WRA Streams MRU that the user had indeed accessed the CD, and in this case he used 4 different PCs to do so. What I needed to do was to see if there was a corroborative method to compliment WRA.

Rather than go through Jesse Hager's PDF in full, I am going to keep to a minimum the necessity to make reference to the PDF. By keeping this to a minimum, hopefully all users of the program should be able to replicate the programs output.

Due to actual Window dimensions I have not been able to incorporate the domain address, but I have included this in the breakdown.

The interesting fact about most LNK file is that they inherit the Creation Time and Last Accessed Date on each relevant folder in the full path. In the LNK on the following page, you can see the path:

C:\Documents and Settings\randy\Local Settings\Temporary Internet Files\Content.IE5 \O1QBCPMR\Rita's clouds.jpg

The LNK files have a makeup similar to C:/Windows (Creation Date and LA) Program Files (Creation Date and LA) Name of Program (Creation Date and LA).

It is quite interesting when the LNK file is examined. References to files and folders within the LNK can on occasions be seen to no longer reside within the path of the LNK, which should arouse curiosity of file activity / movement.

The Last Accessed Time is not stamped in the subfolder structure within the LNK which is why in the output there are no times.

By going to the relevant offset in the LNK file, identifying the Timestamp requires the user to sweep over the first 4 bytes. This is the Creation Date and Last Access Time which defaults to 23:55. The next 4 bytes are the Last Access Date and Creation Time. Also, be aware that you may have to compensate for Time differences when interpreting the output.

LNK Shortcut Analyser

The dates commencing at FO 1Ch are the snapshot of the Creation Time , Last Accessed and Wriiten Dates for the file attributes in question, in this case Rita's Clouds.jpg



LNK Shortcut Analyser

Interesting Note

When I was doing some research work into LNK files I was trying to get hold of as many different OS LNK files to interpret.

I went onto a well known file sharing program and entered the search term 'LNK', and I was overwhelmed at the amount of files that were actually shared. To most it means nothing, however I became aware that some of the LNK files had 'notable' names particularly those with the second character being an '@'

I decided to download several of the LNK files including the 'notable' ones. I proceeded to process them through WFA and saw that the output told me the Computer Users Name, Volume Serial Number and Volume Name for the file in question where it was located at that time of the LNK creation.

Had I browsed the users shared files, I may have established whether they had indeed the 'notable' Files. That said, I think it would be corroborative evidence to suggest that you 'suspect' the path of the files involved, the Volume Serial Number, potential Volume Name, and depending on path (i.e. C/Documents and Settings/**user**/) the User Name, all before you go knocking on doors.

Allan S Hay

November 2005

ash368@btinternet.com