Digital Investigation (2005) **2**, 89—93

Digital Investigation

www.elsevier.com/locate/diin

The joys of complexity and the deleted file[☆]

Geoff H. Fellows

4 Brockhall Road, Flore Northamptonshire NN7 4NG, UK

KEYWORDS
NTFS;
FAT;
Deleted file;
Allocation unit;
SID;
MFT

Abstract This article considers the improved quality of evidence which may be extracted from computers running under modern operating systems and file systems. By way of illustration the author discusses the treatment of deleted files under legacy DOS systems, Windows 9x systems and the NTFS file system, and illustrates the various data artefacts associated with each. It is clear that, although the evidence resulting from more modern systems is more complex, and that analysts require more in-depth training to understand them, the rewards in terms of evidential probity can be considerable, enabling the analyst to produce evidence which in earlier systems was simply not there to be found.
© 2005 Elsevier Ltd. All rights reserved.

A few weeks ago I was pondering the necessity of preparing evidence, required for a criminal trial, to explain deleted files and the workings of the Windows Recycle Bin under NTFS, and after an hour of struggling with the usual problem of trying to explain complex ideas in simple terms I was struck by the thought, ''How much easier this would have been 12 or 15 years ago.''

Then I began to think about other areas of my field of Forensic Computing, which are nowadays orders of magnitude more complex than they used to be. I found that without too much effort I could think of a good many examples.

There are a number of consequences resulting from this. One of those, of course, was exemplified by my immediate problem of preparing an explanation of complicated issues, which the 'lay person' could understand. Another is the amount of training and understanding which Forensic Analysts need these days in order to do their work effectively. That in turn results in the need for an increased expenditure on the part of their employers on training — sometimes that expenditure is forthcoming and sometimes it is not.

It has also widened the gap, I think, between new analysts and experienced analysts, and it has also resulted in many analysts specialising in particular

[☆] This article examines the rewarding nuggets of evidence to be found in the 'deleted' file. In addition it tracks the improvement in the quality of evidence thanks to progressive operating system changes.

E-mail address: geoff@gfellows.demon.co.uk

1742-2876/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2005.04.001

areas of Forensic Computing. These 'specialists' are then called upon quite frequently to support others who, in the context of a particular case, find that that expertise is required.

But then a much happier thought occurred to me. ''*What about the quality of the evidence that we are finding?*'' It is orders of magnitude better and more reliable than it used to be in those halcyon days of the early 1990s, and that is our reward, as analysts, for all the extra work and training, which we are required to undertake.

Deleted Files are a prime example of this reward. Consider the changes which have taken place over the years, and their implications.

Deleted files on a FAT system

In the days of DOS operating systems and FAT-based file systems a deleted file was a simple thing. Once a file was deleted, the first letter of the file name was lost (overwritten with the hexadecimal character E5h) and the areas of the File Allocation Table which used to point to the storage areas on the disk occupied by the file were set to zero, indicating that those allocation units were available for reuse. That was it.

The directory entry for the file concerned was subsequently liable to be overwritten, and the allocation units were also liable to be overwritten, but provided both survived then the likelihood of a successful recovery of the deleted file depended upon a number of factors:

- How many deleted files were there on the disk?
- How fragmented was the file before deletion?
- How big was the file?

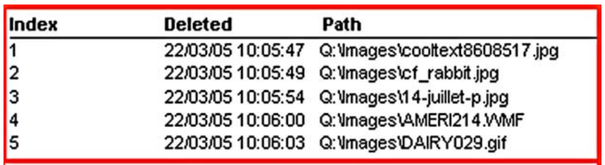
In almost every circumstance on a FAT-based system the recovery of a deleted file from a disk involved a large element of guesswork on the part of the analyst or on the part of the recovery utility used because the actual FAT pointers were lost. Recovering the file was largely a matter of hoping that the sequence of allocation units occupied by the file data could be reliably identified.

Advent of the FAT Recycle Bin

Windows 95, whilst still being a FAT-only based system, introduced the concept of the Recycle Bin, and suddenly things got a bit better.

On a FAT-based disk (even under Windows XP Pro) the actual system folder which constitutes the Recycle Bin is called 'Recycled', and in broad terms it is generally only files which are deleted *by a user* of the computer which are placed in the Recycle Bin. This is an immediate advantage to the analyst, since it helps to distinguish between user-deleted and system-deleted files.

At an evidential level, I always maintain that files which are placed in the Recycle Bin are not deleted files. They are current files on the system, which are merely in a state preparatory to being deleted (whatever the user of the computer might think).



Index	Deleted	Path
1	22/03/05 10:05:47	Q:\Images\cooltext8608517.jpg
2	22/03/05 10:05:49	Q:\Images\cf_rabbit.jpg
3	22/03/05 10:05:54	Q:\Images\14-juillet-p.jpg
4	22/03/05 10:06:00	Q:\Images\AMERI214.WMF
5	22/03/05 10:06:03	Q:\Images\DAIRY029.gif

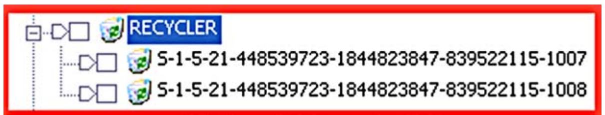
Decoded INFO2 Records

It gets better. The existence of the data in the INFO2 file in the Recycle Bin folders (INFO on some systems) means it is possible for the analyst to acquire evidence about the date and time of the deletion of the file as well as what its original full path was on the disk. There is a logical structure to this. The records in the INFO2 file are in chronological order of deletion.

Of course, once the file is cleared from the Recycle Bin it becomes a deleted FAT file like any other and the INFO2 records are cleared, but even then, because the INFO2 file has an ordered structure of records of 280 bytes in length (or 800 under Windows 2000, XP etc.) it is sometimes still possible in these circumstances to recover deleted INFO2 file data and hence valuable evidence.

NTFS Recycle bins

Under the New Technology File System (NTFS) the evidence gets better still.



```
RECYCLER
    S-1-5-21-448539723-1844823847-839522115-1007
    S-1-5-21-448539723-1844823847-839522115-1008
```

Windows XP Recycler SID's

One reason for this is the fact that the NTFS Recycle Bin (the hidden system folder is actually called ‘Recycler’ on an NTFS disk, rather than ‘Recycled’) is divided up into user Security ID’s (SID’s), and each user’s deleted files are placed into the folder relating to his or her SID. Of course, the perennial issue about ‘Who’s fingers were on the keyboard’ remains, but nevertheless the SID folders can prove very valuable. The SID is divided into three main parts:

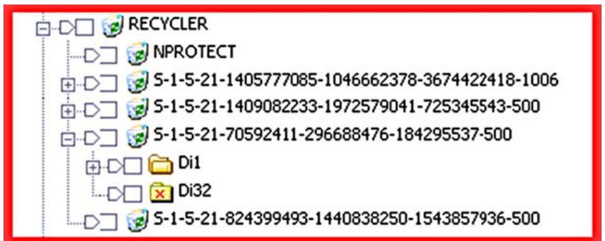
- A ‘Type’ string, which on stand-alone machines at least is usually ‘S-1-5-21.’
- A ‘unique’ computer ID string (in this case 448539723-1844823847-839522115) follows next.
- The number, which indicates the actual user concerned comes last — in this case users 1007 and 1008, as indicated in the Figure, Windows XP Recycler IDs.

The login names associated with these two SID’s can be identified from the Windows Registry files.

Of course, each of the SID folders in the Recycle Bin has its very own INFO2 file, and so the details of file deletion in relation to each user logon name is available as well.

I have come across another interesting circumstance relating to this topic. I was given a ‘LaCie’ 60 GB NTFS formatted removable USB/FireWire hard drive to examine.

The Microsoft Knowledge Base tells us that ‘removable media’ is not given a Recycle Bin, but the definition of what is removable media and what is not is rather loose, saying merely that it is ‘Media which is easily removed from the computer.’



```
RECYCLER
  NPROTECT
  S-1-5-21-1405777085-1046662378-3674422418-1006
  S-1-5-21-1409082233-1972579041-725345543-500
  S-1-5-21-70592411-296688476-184295537-500
    Di1
    Di32
  S-1-5-21-824399493-1440838250-1543857936-500
```

‘LaCie’ Drive Recycler Artefacts.

However, the ‘LaCie’ drive did have a Recycle Bin, and so it clearly did not fall within the above loose definition. What was more interesting, though, was the SID folders contained in the Recycle Bin illustrated in the Figure, ‘LaCie’ Drive Recycler Artefacts. It can be seen that in the past it was connected to four different computers, (evidenced by the Machine ID numbers, and all of them identifiable with a little basic detective work), and whilst connected to each machine, files had been deleted from the ‘LaCie Drive’ contents. On three of the machines the user was ‘Administrator’ (User ID ‘500’) and it transpired that on the fourth he also had Administrator privileges. The user could, of course, have emptied these Recycle Bin artefacts, but the fact is that he could only ‘see’ and hence clear those files in the Recycle Bin which had been deleted *when the drive was re-connected to the matching machine*. Therefore, he was unaware that there was much more data to be found. This misconception and its result led directly to a guilty plea and a conviction.

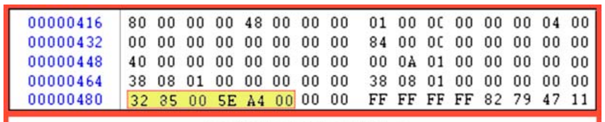
NTFS extents pointers

Can it get better still? Yes, it can. The regime used by NTFS to point to file data out on the disk is quite different from that used in FAT. This results, in broad terms, in recovered deleted files from an NTFS volume being much more reliable, evidentially speaking, than those recovered from a FAT volume.

Under NTFS the central index of data on the disk or volume is kept in a system file, the Master File Table (MFT). This hidden system file is essentially a relational database of the disk content, and contains an entry in it (usually 1024 bytes long) for each file and folder on the disk.

The MFT records are themselves divided into a number of separate data areas which are called ‘Attributes’ (not to be confused with DOS type file attributes — Read Only, Hidden etc.). The ‘Data Attribute’ in an MFT record points to the data on the disk, but it does so using an ‘Extents Pointer’.

As you can see I have illustrated a typical file MFT record Data Attribute. The attribute starts with the header string 80h 00h 00h 00h, and the logical and physical file sizes are included in the attribute as well (as little-endian 64 byte integers — logical file size is 67,640 bytes (offset 472 in the screenshot) and the physical file size is 68,096 bytes (offset 456)).



```
00000416  80 00 00 00 48 00 00 00  01 00 0C 00 00 00 04 00
00000432  00 00 00 00 00 00 00 00  84 00 0C 00 00 00 00 00
00000448  40 00 00 00 00 00 00 00  00 0A 01 00 00 00 00 00
00000464  38 08 01 00 00 00 00 00  38 08 01 00 00 00 00 00
00000480  32 85 00 5E A4 00 00 00  FF FF FF FF 82 79 47 11
```

An MFT Record Data Attribute.

I have highlighted the extents pointer, starting at offset 480. This simple and elegant device is all that is required to specify where this (contiguous) file lies on the disk and how many allocation units it occupies. It decodes as follows:

Extents header	Number of allocation units in the data run (little-endian signed)	Logical allocation unit number where file starts on the volume
32	85 00	5E A4 00

The two nybbles of the extents header, added together, makes five ($2 + 3 = 5$), which means that the next five bytes following the header are required to be interpreted. Those five bytes are 85h 00h 5Eh A4h 00h.

The fact that the second nybble of the extents header is a 2 means that the next two bytes are required to define the number of allocation units in the data run. This little-endian signed integer translates into decimal as $+133$, meaning that the data run is 133 allocation units (on this disk synonymous with 133×512 byte sectors). This 'ties in' with the physical size of the file (68,096 bytes), since $512 \times 133 = 68{,}096$ bytes.

The first nybble of the extents header is a 3, which means that the remaining 3 bytes describe the starting logical allocation unit on the volume for the file. It is also a signed little-endian integer. In this case 5Eh A4h 00h translates as $+42{,}078$, meaning that the file starts at logical allocation unit 42,078 on the disk. It is as simple as that.

If the files were fragmented there would be a series of two or more extents pointers, but each one would work in the same way. The only difference is that the start of run pointer for subsequent extents pointers is relative to the start of the one before it, and it might be a negative value – i.e. relatively further back up the disk from the last fragment.

What has all this to do with deleted files? When a file is deleted on an NTFS volume two main events occur as far as the file system is concerned. Firstly, the MFT record for the file is marked as relating to a deleted file (the bytes at MFT record offset 22 and 23 are changed from 01h 00h to 00h 00h (in the case of a file). Secondly, a system file called $Bitmap is updated to show that the allocation units occupied by the file are available for reuse by the system.

Nothing in the MFT record's data (or other) attributes is disturbed or altered (unless the MFT record comes to be overwritten itself). That in turn means that when the recovery of a deleted NTFS file is made, the analyst (or the recovery software used) knows *precisely where that file used to reside on the disk*. There is no equivalent needed here to the 'guessing' of FAT record pointers, and provided that the data on the disk has not been overwritten by another file then the deleted file will be recovered reliably and to a much higher evidential standard than under FAT. Recovering fragmented files is not such a difficult issue either, since all of the extents pointers for the fragments, however many there are, will still be present in the MFT data attribute.

And finally…

Even where the data has been partially overwritten on an NTFS volume, but the MFT record still survives, the analyst may still be able to recover and to produce useful evidence.

Consider the circumstance where the file above, occupying 133 allocation units, is deleted, and then a smaller file later overwrites the first, say, 10 allocation units but the MFT record for the deleted file is not overwritten.

The extents pointer for the old (deleted) record and the pointer for the new (current) record will both point to the same place – logical allocation unit number 42,078, and the analyst will know that the deleted file was not fragmented. Therefore, he or she *may* find the remainder of the deleted file in logical allocation unit numbers 42,088 through 42,211, provided they have not also been overwritten.

A physical inspection of the data in those sectors may well correlate strongly with the information about the file in the surviving deleted MFT record. That record contains not only the logical and physical size of the file, but also the file dates and times, the name(s) of the file (both long and short), the file's DOS attributes and so forth. In some cases it will also indicate who is the owner of the file and who has 'rights' in relation to it (coupled with the system file $Secure and its alternate data streams).

Consider the following trail of evidence: The deleted file's MFT record shows that it was called 'Kidnap Demand.Doc.' The data in the remaining clusters has the format of an OLE2 Microsoft Word document. In accordance with the file name the textual content appears clearly to relate to a kidnap demand for ransom. In addition the old file ends, in the last sector, in exactly the right place to accord with the logical file size found in the MFT record. As a result the analyst would arguably be in a strong evidential position to maintain that a Microsoft Word document called

Kidnap Demand.doc did exist on the computer and that the partially overwritten fragment which he/she has recovered represents a substantial part of that original document.

So, all in all, because of the joys of complexity, our evidence nowadays about deleted files is much better than ever it used to be in the 'Good Old Days', but producing it requires a lot more education of, and understanding by, the analyst involved in examining the data.

Provided analysts are properly supported and funded by an organization which understands their needs and the issues involved, the evidence which can be produced in this and other similar areas looks set to improve.

Geoff Fellows is founder of The LG Training Partnership. Prior to its establishment he was in the Northamptonshire police in the UK. He is chairman of the F3 and is a member of the UK Digital Evidence Group (DEG).