



## sanderson forensics

### ***NTFS Compression - a forensic view***

By: Paul Sanderson, Sanderson Forensics - October 2002

#### ***Introduction***

One of the little discussed, forensically speaking, 'features' of the NTFS file system is that of file compression.

File compression can be turned on at various levels; a user can choose to compress individual files, a whole folder or the whole of a drive. Once compression is turned on for a folder or drive then any new files added to the folder/drive are automatically compressed without user intervention.

As of writing none of the commercially available computer forensic toolkits support compressed file recovery, the implications of this are discussed below.

#### ***Scope***

This white paper will discuss the basic concepts of NTFS file level compression and the impact it has on a forensic analysis. We will also cover the difficulties associated with forensic examination of deleted compressed files. It will not cover the specifics of the compression algorithm nor will it cover the exact mechanics of how a compressed file is allocated and how the operating system tracks the file.

#### ***NTFS compression***

##### **How it works**

A primary design goal of any modern operating system is speed of access. To facilitate this NTFS compression works as follows:

- Each file is split into compression units of 16 clusters.
- Within a compression unit compression is cluster based
- Each cluster is independently compressed and can be decompressed independently of any other cluster within the run.

An example (for the purposes of example the cluster size is assumed to be 4096 Bytes):

A file that currently occupies say 100 clusters is compressed.

When the first 16 clusters are compressed it is found that the compressed data will occupy 5 clusters (4 clusters and a partial cluster). This data is saved to disk and 5 clusters are allocated to the file

The second 16 clusters, when compressed, do not result in any space savings and so are saved to disk uncompressed i.e. all 16 clusters are saved and allocated to the file – it is now 21 clusters in length (5 + 16).

The third compression unit compresses to 15 clusters – our file is now 36 clusters in length.

Compression proceeds in this way until all 100 clusters of the file are accounted for.

Within a compression unit the data is stored as follows:

Each cluster is independently compressed, so the first 4096 (1 cluster) bytes of data are compressed. If the data is compressible, i.e. a space saving is made, then the data is saved as a two byte word indicating the compressed length of the following data and then the data follows.

If the compression factor is so low as to not save at least one clusters worth of storage in a compression run then that compression run is saved uncompressed. If the data is not compressible then it is saved uncompressed with a length word of 0xFFFF

What happens when a file needs to be uncompressed?

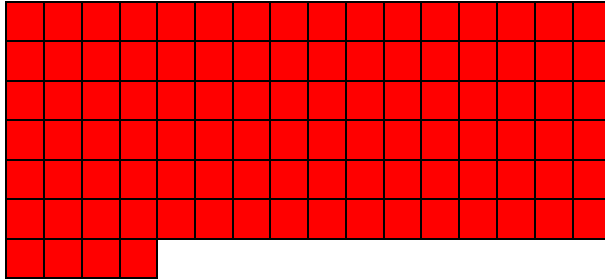
The MFT is read:

- If the allocated space for the first 16 clusters (compression unit) is 16 clusters then the compression unit has been stored 'as is' i.e. it has not been compressed.
- If the allocated space is less than 16 clusters then some compression has taken place, so each cluster is decompressed in turn:
- The first two bytes of the compression unit are read and this number of bytes decompressed to get the first cluster. The second length follows immediately after the first compressed cluster – this is read and the data decompressed.....

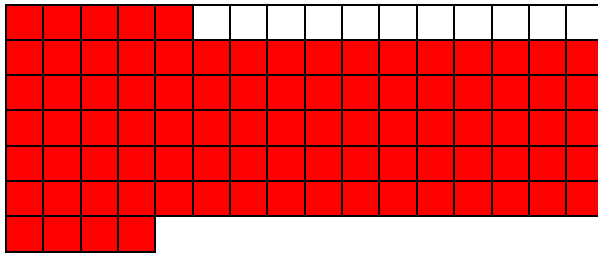
## What does a compressed file look like on disk

For our 100 cluster file we may see (assuming the file is stored in contiguous clusters):

The file is initially allocated clusters 0 through to 99

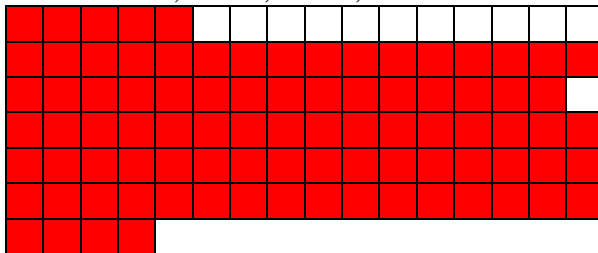


When the file is compressed the compression engine will read the first compression unit (clusters 0-15) and compress them to 5 clusters – so our file now occupies clusters 0-4 and 16-99



The second compression unit is not compressible so this is written to disk as is.

The third compression unit compresses to 15 clusters so we now have allocation of:  
Clusters: 0-4, 16-31, 32-46, 48-99



And so on....

This reflects itself as a file that appears to have runs of blank (NULL) data at the end of most 16 cluster runs.

On the face of it, it would seem logical for the compressed data from the second compression unit to be saved to disk immediately after the first compression unit ends. It is my belief that Microsoft has chosen this method for reasons of efficiency. By leaving the saved clusters within a compression unit initially empty means that if data is added to the first compression run then it just needs to be expanded into the free space that was left after initial compression. The operating space will only use this freed up space to store another file, should space become a premium.

## Implications

There are a number of implications for the forensic examiner, from the above it can be seen that:

A compressed file can consist of both compressed and uncompressed data

A file such as a word document containing lots of white space and other compressible data will usually yield good compression

A file such as a JPEG that already has a highly compressed format will yield poor compression. In fact jpg's usually have a compressed first cluster (this normally contains compressible manufacturers info) followed by uncompressed clusters and compression units.

If a document exists on disk in a compressed form then it stays in its compressed form when deleted, therefore searching a disk for a keyword that only exists within the compressed deleted file will not yield a result.

As length information precedes any data within a compression unit a search for a file header on a compressed file (or deleted file) will likely be unsuccessful.

Decompressing a file without recourse to the files MFT record may not be possible i.e. The only indication of whether a compression unit has been compressed is within the MFT. This means that a file 'carver' may not be successful in a) extracting the compressed file and b) decompressing it.

If your forensic tool does not support compression then a compressed file could appear to be longer than it is, i.e.:

Consider a file that is 32K bytes in length (uncompressed) when this is compressed it will occupy, say, 5 clusters. The MFT though reports the uncompressed file length and as 32KB will not fit uncompressed into 5 clusters (20KB) then the area beyond the end of the compressed file will be reported as belonging to the file when in fact it should be slack space.

Your forensic tool may report that the allocation for the file is incorrect, i.e. only 5 clusters are allocated to a file that should (based on the uncompressed length in the MFT) require 8 clusters.

## ***Recommendations***

If your chosen forensic tool does not support compressed files then thought should be given to mounting a clone of the drive on a system that does support compression.

I believe that currently there is no software tool that will allow for the recovery of deleted compressed files.

## ***About the author***

Paul Sanderson, has extensive experience within the field of computer forensics, having been responsible for the development of some of the most widely used software investigation tools. Paul has worked with and for some of the most respected organisations in this field and has given written and oral evidence in court on a number of occasions, Paul has also provided expert assistance in numerous other cases which has often helped achieve early settlement

Working with computers in a Computer Forensic/Data Recovery capacity since early 1993. The first six years were with “Vogon International” (formerly AuthenTec, formerly S&S International) where he was responsible for the development of the Computer Forensic software suite, at the time this was used by the majority of the UK police forces and investigating authorities. His time at Vogon culminated as the General Manager of the Munich office where he was responsible for developing both the Computer Forensic and Data Recovery business.

On leaving Vogon Paul worked for a year with “Network International”, a corporate investigations company in London. Paul then set up his own consultancy in April 2000. He have advised, amongst others, a number of UK and the Northern Ireland Police Forces, Customs & Excise, the Serious Fraud Office and a number of large UK commercial organisations. Paul writes his own Computer Forensic software and sells to a number of organisations including various UK and US police forces, and investigative bodies.

<http://www.sandersonforensics.com>