Incident Response with NTFS INDX Buffers – Part 2: The Internal Structures of a File Name Attribute

By Jeff Hamm on September 26, 2012

By Jeff Hamm & William Ballenthin

Last week on M-Unition, Willi and I published the first post in the NTFS INDX Buffers series. The first post explained how to extract an INDX attribute using various tools. In part two of the series, we detail the binary structure of a file name attribute. The file name attribute is used to track an INDX buffer's content.

Part 2: The Internal Structures of a File Name Attribute

A filename attribute in the NTFS file system tracks metadata associated with a single filename of one file. Multiple filename attributes can exist for any file – such as a short file name alias. This metadata includes the filename, timestamps, physical file size, and the logical file size. Filename attributes are always resident in the $MFT (Master File Table) as a sub-structure of an $MFT file record. These attributes have a dynamic size in the $MFT because path components in NTFS range from 1 to 255 characters.

The Master File Table

To understand the role of the filename attribute, we first need to review the structure of the $MFT. The $MFT is a special NTFS file that organizes the rest of the files on the file system. It is made up of metadata and defines the relations between all directories and files. In some sense, the $MFT file is the essence of the NTFS file system.

NTFS structures the $MFT into small segments, called records – each of which describes one file or directory. When these records correspond to files, a record indicates where to find the data content of the file. When these records correspond to directories, the record tracks the contents of the directory. In both cases an $MFT record stores metadata in attributes.

Examples of attributes include standard information attributes (type 0×10), filename information attributes (type 0×30), and data attributes (type 0×80). While an $MFT record is 1024 bytes in length, attributes are variably sized. They sit consecutively within an $MFT record.

Filename Attributes

Filename attributes are a particular type of $MFT record attribute. They track information associated with a single filename for one file. In NTFS, a file may have more than one filename. For example, most files have an 8.3 filename (such as "DOCUME~1") in addition to a Unicode filename (such as "Documents and Settings"). This means each $MFT record often has more than one filename attribute.

The attribute stores modified, accessed, changed, and created/birth (MACb) timestamps that relate to the filename value. These timestamps are updated independently from the standard information attribute

timestamps, which change when a user accesses or modifies the contents of a file. Filename attribute timestamps change when the filename for a file is updated, such as when a user renames a file from "Secret Plans of World Domination.docx" to "Nothing to See Here.docx".

Forensic investigators are encouraged to review filename attribute data because it offers alternate and additional data points for interesting files. Because the filename attribute contains its own set of MACb timestamps, and there may even be many filename attributes for one interesting file, an investigator may generate a much more robust timeline of actively. "Time-stomping", or modifying MACb timestamp information for anti-forensics, is easy for standard information attribute timestamps. However, filename timestamps are tricky to persuasively modify because the Windows API does not expose the necessary functions. Forensic investigators often review all timestamps as a set to identify anomalies, and filename timestamps are critical in this process.

Besides timestamps and filenames, a filename attribute contains a file offset to the $MFT record of the containing directory. An investigator may use these pointers to reconstruct the directory hierarchy of a file system. For example, both analyzeMFT.py (http://www.integriography.com/) and MFTINDX.py (https://github.com/williballenthin/INDXParse) use this technique to process an $MFT and report the


```
    typedef struct {
/* 0x00 */uint8_t parent_reference[6];
/* 0x06 */uint8_t
parent_sequence_number[2];
/* 0x08 */uint8_t created_time[8];
/* 0x10 */uint8_t modified_time[8];
/* 0x18 */uint8_t changed_time[8];
/* 0x20 */uint8_t accessed_time[8];
/* 0x28 */uint8_t logical_size[8];
/* 0x30 */uint8_t physical_size[8];
/* 0x38 */uint8_t flags[8];
/* 0x40 */uint8_t filename_length;
/* 0x41 */uint8_t filename_namespace;
/* 0x42 */uint8_t filename;
    } ntfs_attr_fname;
```

Figure 1: TSK File Name Attribute Code

Offset	Length	Field
0x00	6	Parent Directory $MFT Record Number
0x06	2	Parent Directory Sequence Number
0x08	8	Created (Birth) Time
0x10	8	Last Written Time
0x18	8	Record Modified
0x20	8	Last Accessed Time
0x28	8	Logical Size
0x30	8	Physical Size
0x38	8	Flags (Read Only, Hidden, etc)
0x40	1	File Name Length (Unicode characters)
0x41	1	Namespace
0x42	*	File Name

Table 1: File Name Attribute

Conclusion

Filename attributes are a structure of NTFS that contain forensically relevant metadata. Investigators should use filename attribute timestamps to augment their timelines and identify anomalies. Dead-box forensic tools also use these attributes to reconstruct the folder structure with which a suspect or victim interacted. Reviewing and understanding the binary structure of these key artifacts enables you to grok existing techniques, and potentially develop novel solutions to finding evil. Furthermore, NTFS indices use the same structure as filename attributes to store information – we'll readdress this in Part four of this series.

Please note we have updated last week's post to include a fourth tool, Mandiant Intelligent Response® (MIR). The MIR agent v.2.2 has the ability to extract INDX records natively. We recommend reviewing the revised post to learn how to use MIR with INDX records.

Tags: forensics, INDX buffers, Intelligent Response, MIR, NTFS, NTFS INDX Buffers, SleuthKit

Category: The Lab

OLDER ENTRY NEWER ENTRY

Incident Response with NTFS | Accelerate Your Incident | SANS WhatWorks Summit in

INDX Buffers – Part 1: Extracting an INDX Attribute

Response with the latest version of Mandiant Intelligent Response® (MIR®)

Forensics and Incident Response

Comments

Leave a Comment

Name (required)

Email (required)
Your email will never be shared.

URL

Submit Your Comment

Notify me of follow-up comments by email.

Notify me of new posts by email.

TEL +1 (703) 683-3141 TOLL FREE (800) 647-7020

DC HEADQUARTERS: 2318 MILL ROAD #500, ALEXANDRIA, VA 22314 (GOOGLE MAP) OTHER LOCATIONS

FIND US ON THE WEB: