get-file-info: A tool for inspecting NTFS MFT records

`get-file-info` is a tool for inspecting NTFS MFT records. An analyst can use it to review the metadata associated with a file path, including timestamps, attributes, and data runs. You'll find the tool useful to challenge or confirm artifact interpretations and recover evidence of deleted files.

Download

`get-file-info` is a component of the INDXParse suite of tools used for NTFS analysis. All INDXParse tools are free and open source. The source for `get-file-info` is hosted on Github here.

Highlights

Timelines `get-file-info` automatically generates a timeline of all timestamps identified in the target MFT record. These include timestamps from the STANDARD INFORMATION attribute, FILENAME attribute, and resident directory index entries. You'll find this quickly highlights timestomping and scopes a malware infection.

Data formatting `get-file-info` parses NTFS attributes and presents them in a human readable format. Since the tool simply formats raw data, it can help you challenge or confirm the interpretation of data by commercial tools. And, when you encounter resident data, the tool will display a hex dump of its contents.

Strings `get-file-info` extracts ASCII and UTF-16LE strings from both the active and slack spaces within an MFT record. Wide strings from slack space often identify metadata from previously deleted files and directories.

Free All INDXParse tools are free and open source. Forensic practitioners drive the development by contributing ideas, bug reports, and patches. Since the source is in the open and covered by a liberal licnse, you'll never have to worry about the tools disappearing.

Command line driven, text interface `get-file-info` is a tool that executes from the command line using a Python 2 interpreter. Since it produces text output, you'll find it easy to script and integrate with your workflow.

Installation

`get-file-info` is part of the INDXParse suite of tools that are distributed together. To acquire INDXParse, download the latest ZIP archive from here or use `git` to clone the source repository:

```
1 git clone https://github.com/williballenthin/INDXParse.git
```

`get-file-info` depends on a few freely available Python modules. You should install these using `pip`, as described here. The modules are:

- argparse
- jinja2
- python-progressbar

You can install them all in one go like this:

```
1 pip install argparse jinja2 python-progressbar
```

Usage

`get-file-info` is a Python script that should be run from the command line. It accepts two command line parameters: `mft` and `record_or_path`.

- The first argument is the path to a raw MFT file previously acquired. Due to access restrictions imposed by Microsoft Windows, you cannot run this tool against the MFT of a live system.
- The second argument identifies which MFT record you intend to inspect.
    - The identifier may be a number, in which case the tool displays the record with the exact record number. This involves an array lookup, so the script will complete quickly.
    - If the identifier is a string, then the tool interprets this as a path, and attempts to find the record associated with the path. This may involve rebuilding the entire file system tree, so script execution is not guaranteed to be fast. From personal experience, providing a path identifier often takes between five and 30 seconds.

Here's an example of a user inspecting the MFT record zero (the record for the MFT itself):

```
1 python get_file_info.py /evidence/case001/CMFT 0
```

Here's an example of a user inspecting the MFT record for the path `C:\WINDOWS\Temp`. Note, the does not know the volume name, so the prefix "C:" is not provided.

```
1 python get_file_info.py /evidence/case001/CMFT "\WINDOWS\Temp"
```

Sample Output

Here's a sample listing of the tool executed against the MFT record associated with a system's "%TEMP%" directory:

```
1  Git/INDXParse - [master●] » python get_file_info.py MFT.copy0 72
2  MFT Record: 72
3  Path: \.\WINDOWS\Temp
```

```
Metadata:
  Active: 1
  Type: directory
  Flags:
  $SI Modified: 2012-07-05 23:25:01.562498
  $SI Accessed: 2012-07-05 23:25:01.562498
  $SI Changed: 2012-07-05 23:25:01.562498
  $SI Birthed: 2010-12-16 11:41:33.312498
  Owner ID: 0
  Security ID: 686
  Quota charged: 0
  USN: 0
Filenames:
  Type: WIN32 + DOS 8.3
    Name: Temp
    Flags: has-indx
    Logical size: 0
    Physical size: 0
    Modified: 2010-12-16 11:41:33.312498
    Accessed: 2010-12-16 11:41:33.312498
    Changed: 2010-12-16 11:41:33.312498
    Birthed: 2010-12-16 11:41:33.312498
    Parent reference: 28
    Parent sequence number: 1
Attributes:
  Type: $STANDARD INFORMATION
    Name: <none>
    Flags: has-indx
    Resident: True
    Data size: 0
    Allocated size: 0
    Value size: 72
  Type: $FILENAME INFORMATION
    Name: <none>
    Flags: has-indx
    Resident: True
    Data size: 0
    Allocated size: 0
    Value size: 74
  Type: $INDEX ROOT
    Name: $I30
    Flags: has-indx
    Resident: True
    Data size: 0
    Allocated size: 0
    Value size: 56
  Type: $INDEX ALLOCATION
    Name: $I30
    Flags: has-indx
    Resident: False
    Data size: 45056
    Allocated size: 45056
    Value size: 0
    Data runs:
      Offset (clusters): 7289710 Length (clusters): 11
  Type: $BITMAP
    Name: $I30
```

```
61     Flags: has-indx
62     Resident: True
63     Data size: 0
64     Allocated size: 0
65     Value size: 8
66 INDX root entries: \<none\>
67 INDX root slack entries: \<none\>
68 Timeline:
69   2010-12-16 11:41:33.312498    birthed     $SI     Temp
70   2010-12-16 11:41:33.312498    birthed     $FN     Temp
71   2010-12-16 11:41:33.312498    accessed    $FN     Temp
72   2010-12-16 11:41:33.312498    modified    $FN     Temp
73   2010-12-16 11:41:33.312498    changed     $FN     Temp
74   2012-07-05 23:25:01.562498    accessed    $SI     Temp
75   2012-07-05 23:25:01.562498    modified    $SI     Temp
76   2012-07-05 23:25:01.562498    changed     $SI     Temp
77 ASCII strings:
78   FILE0
79 Unicode strings:
80   Temp
81   $I300
82   $I30
83   $I30
84   37.tmper
85   MPC4B.tmper
86   MPC5F.tmper
87   $I30
88   $I30?
89   ~DFF2A7.tmp
```

- williballenthin.com
  - INDX parsing
  - python-registry module
  - Shellbags analysis
  - python-evtx module
- GitHub
- Twitter

- Curated RSS
- search
- Contact