TZWorks® Microsoft Office Backstage (*bs*) Parser Users Guide

Abstract
bs is a standalone, command-line tool that parses the Microsoft Office 2016 Backstage artifacts. The Backstage artifact include entries of files and folders with their last modified timestamp. The results are displayed in a CSV type format where one record is displayed per line. This tool has binary versions that run in Windows, Linux and OS-X.

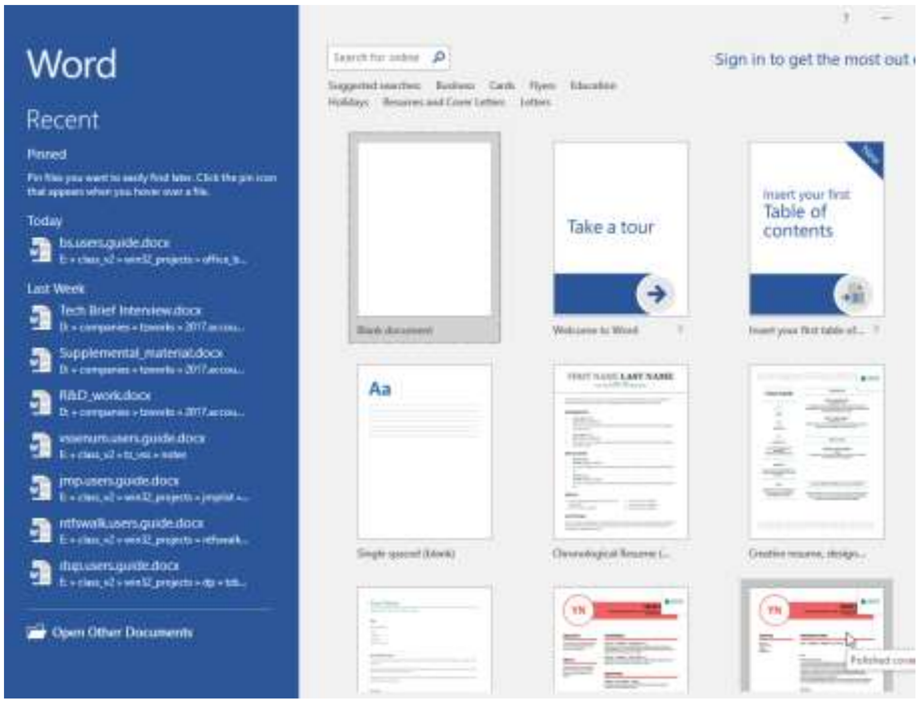
Table of Contents

TZWorks® Microsoft Office Backstage (*bs*) Parser Users Guide

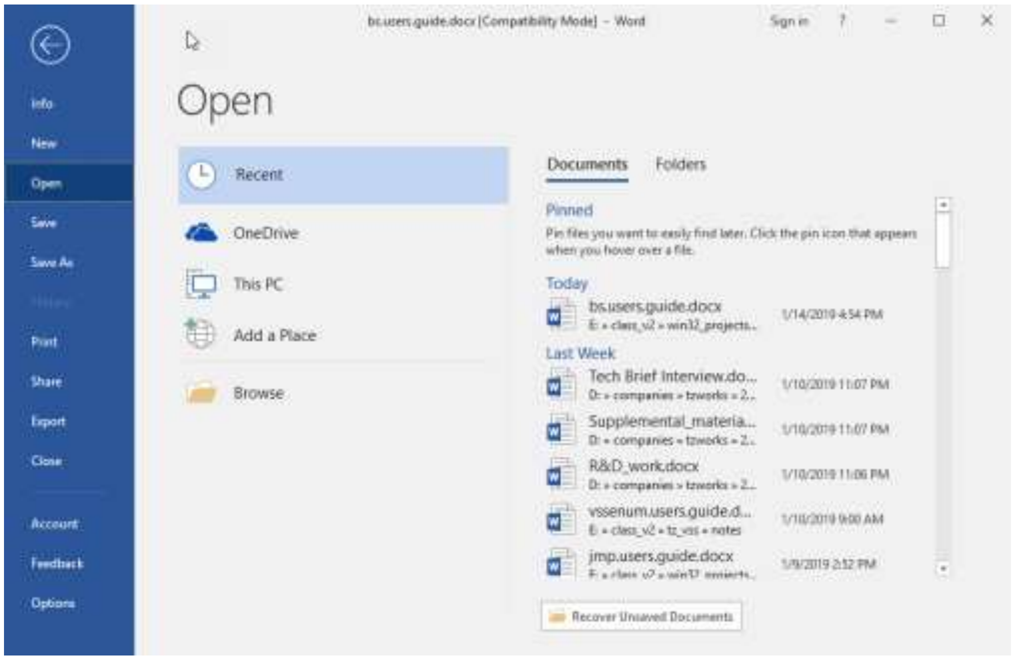
Copyright © *TZWorks LLC*
Webpage: http://www.tzworks.com/prototype_page.php?proto_id=44
Contact Information: info@tzworks.com

1 Introduction

With the newer versions of Microsoft (MS) Office programs, when you first start Office you will be presented with the Backstage view. From this view, you can create a new document (using a pre-created template) or open an existing file. One can also see the most recently used files listed on the left side of the view.



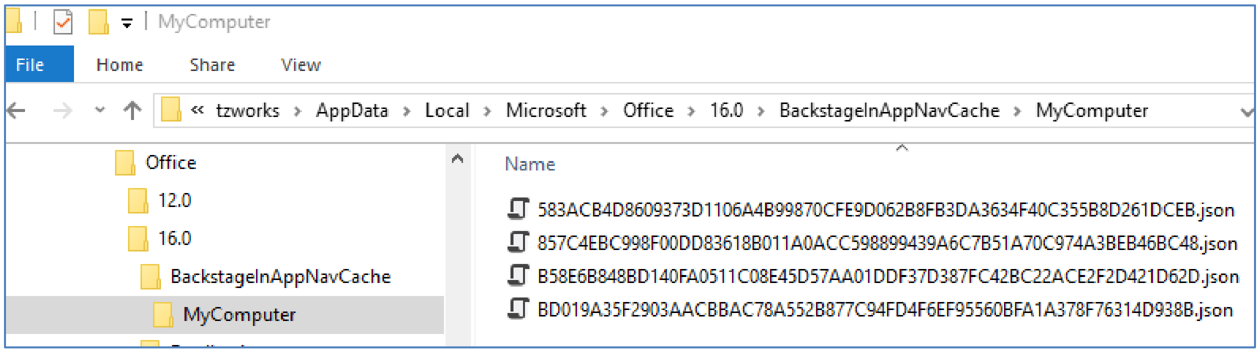
If you are currently viewing or editing a file, and you want to go back to the Backstage view, one just selects the ***File*** tab.



In order for MS Office to render the history data, it makes use of some persistent information stored on the computer.   For MS Office 2016, this file history data is contained in a new set of files located in the *MyComputer* folder or other folders that designate remote shares.

C:\Users\<acct>\AppData\Local\Microsoft\Office\16.0\BackstageInAppNavCache\[MyComputer or other remote dir]

The files residing in this directory can be either delimited text or json formatted text. These files have long names that consist of a 64-character string. The string is actually a representation of a 32-byte hexadecimal hash which results from the computation of certain contents of the artifact file that relate to a primary directory. This, in turn, allows unique names files to be generated. Below is an example of a few of the JSON type files on one of our test computers.

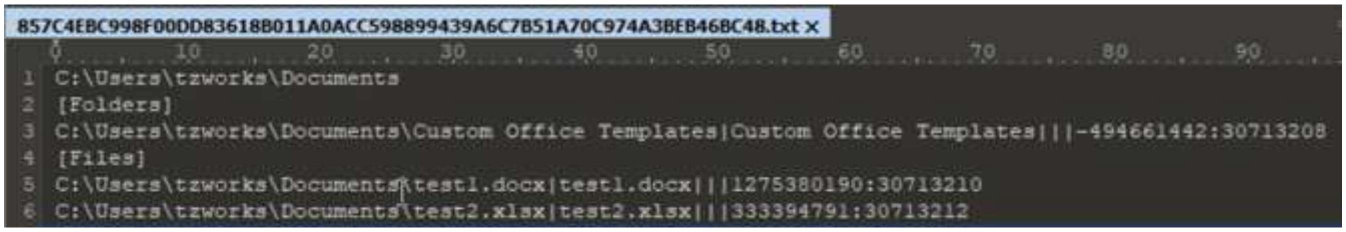

MyComputer
File
Home
Share
View
« tzworks › AppData › Local › Microsoft › Office › 16.0 › BackstageInAppNavCache › MyComputer
Office
12.0
16.0
BackstageInAppNavCache
MyComputer
Name

583ACB4D8609373D1106A4B99870CFE9D062B8FB3DA3634F40C355B8D261DCEB.json
857C4EBC998F00DD83618B011A0ACC598899439A6C7B51A70C974A3BEB46BC48.json
B58E6B848BD140FA0511C08E45D57AA01DDF37D387FC42BC22ACE2F2D421D62D.json
BD019A35F2903AACBBAC78A552B877C94FD4F6EF95560BFA1A378F76314D938B.json

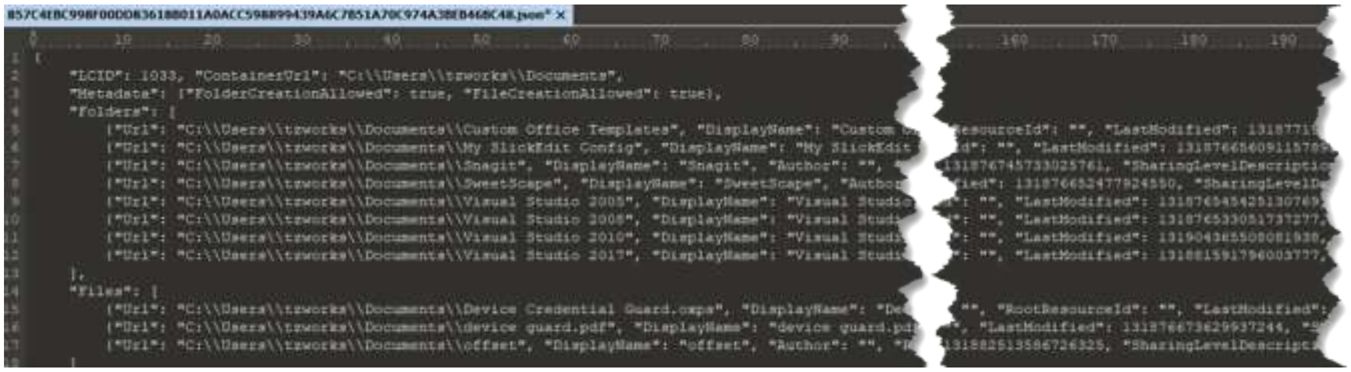
When parsing the data in backstage files, of interest to the analyst is the data that contains references to file and folder paths (both local and remote), each timestamped with the last modified time. So, while the records identify files and folders used in the past, it doesn't necessarily mean they still exist on the system. Therefore, this data can be good in identifying user activity in conjunction with certain files even after these same files may have been deleted or moved elsewhere.

2 Internals of the Artifact File

As mention earlier, the Backstage artifact data consists of text data. The older format has records where each field is delimited by the pipe character. The newer format has records that use the JSON (JavaScript Object Notation). The interesting fields that are populated for most records are the: *Url (folder)*, *Displayname (file)*, and *LastModified* timestamp. There are other fields, but they are not populated as often or not at all depending on your configuration, and include: *Author*, *ResourceID,* and a few others.   Below are samples of the two formats encountered; the first screenshot shows records delimited by a pipe character and the second one is JSON. For the second one, we took the liberty of formatting the JSON output so it could be shown in this document; typically, the Backstage artifact renders JSON data without any CRLFs, and consequently, would look like a single very long line.


857C4EBC998F00DD83618B011A0ACC598899439A6C7B51A70C974A3BEB468C48.txt

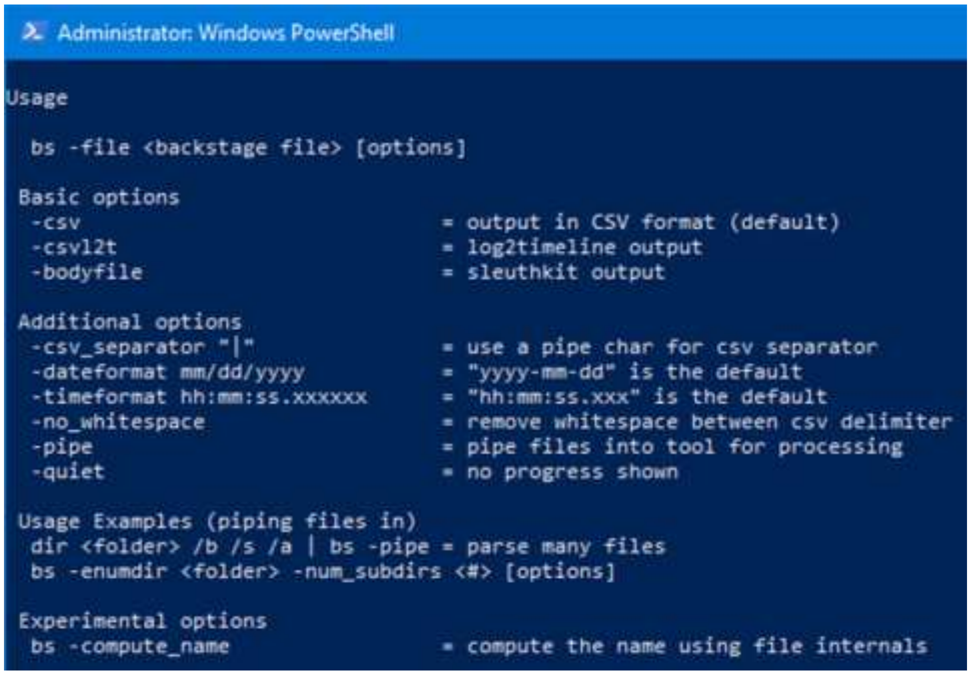
```
C:\Users\tzworks\Documents
[Folders]
C:\Users\tzworks\Documents\Custom Office Templates|Custom Office Templates|||-494661442:30713208
[Files]
C:\Users\tzworks\Documents\test1.docx|test1.docx|||1275380190:30713210
C:\Users\tzworks\Documents\test2.xlsx|test2.xlsx|||333394791:30713212
```


857C4EBC998F00DD83618B011A0ACC598899439A6C7B51A70C974A3BEB468C48.json*

```
{
    "LCID": 1033, "ContainerUrl": "C:\\Users\\tzworks\\Documents",
    "Metadata": {"FolderCreationAllowed": true, "FileCreationAllowed": true},
    "Folders": [
        {"Url": "C:\\Users\\tzworks\\Documents\\Custom Office Templates", "DisplayName": "Custom O    esourceId": "", "LastModified": 1318771
        {"Url": "C:\\Users\\tzworks\\Documents\\My SlickEdit Config", "DisplayName": "My SlickEdit    d": "", "LastModified": 131876656091157
        {"Url": "C:\\Users\\tzworks\\Documents\\Snagit", "DisplayName": "Snagit", "Author": "",    131876745733025761, "SharingLevelDescriptio
        {"Url": "C:\\Users\\tzworks\\Documents\\SweetScape", "DisplayName": "SweetScape", "Author    fied": 131876652477924550, "SharingLevelD
        {"Url": "C:\\Users\\tzworks\\Documents\\Visual Studio 2005", "DisplayName": "Visual Studio    ": "", "LastModified": 131876545425130769,
        {"Url": "C:\\Users\\tzworks\\Documents\\Visual Studio 2008", "DisplayName": "Visual Studi    ": "", "LastModified": 131876533051737277,
        {"Url": "C:\\Users\\tzworks\\Documents\\Visual Studio 2010", "DisplayName": "Visual Studi    ": "", "LastModified": 131904385508081936,
        {"Url": "C:\\Users\\tzworks\\Documents\\Visual Studio 2017", "DisplayName": "Visual Studi    ": "", "LastModified": 131881591796003777,
    ],
    "Files": [
        {"Url": "C:\\Users\\tzworks\\Documents\\Device Credential Guard.oxps", "DisplayName": "De    "", "RootResourceId": "", "LastModified":
        {"Url": "C:\\Users\\tzworks\\Documents\\device guard.pdf", "DisplayName": "device guard.pd    ", "LastModified": 131876673629937244, "S
        {"Url": "C:\\Users\\tzworks\\Documents\\offset", "DisplayName": "offset", "Author": "", "R    131882513586726325, "SharingLevelDescript
    ]
```

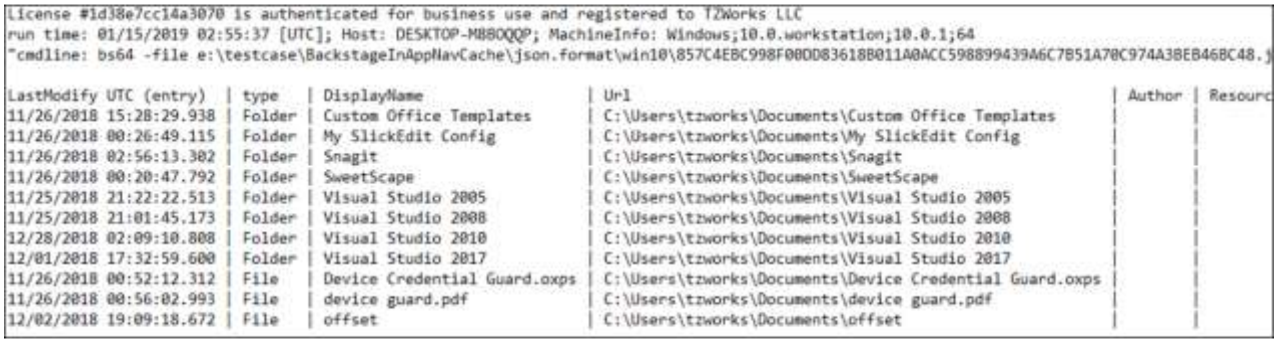
3 How to Use *bs*

The screen shot below shows all the options available for this tool.



```
Usage

 bs -file <backstage file> [options]

Basic options
 -csv                          = output in CSV format (default)
 -csvl2t                       = log2timeline output
 -bodyfile                     = sleuthkit output

Additional options
 -csv_separator "|"            = use a pipe char for csv separator
 -dateformat mm/dd/yyyy        = "yyyy-mm-dd" is the default
 -timeformat hh:mm:ss.xxxxxx   = "hh:mm:ss.xxx" is the default
 -no_whitespace                = remove whitespace between csv delimiter
 -pipe                         = pipe files into tool for processing
 -quiet                        = no progress shown

Usage Examples (piping files in)
 dir <folder> /b /s /a | bs -pipe = parse many files
 bs -enumdir <folder> -num_subdirs <#> [options]

Experimental options
 bs -compute_name              = compute the name using file internals
```

The most basic option is to parse an individual file. One does this by using the ***-file <argument>*** option. This takes the artifact path/file as an argument. The tool can sense, based on the file contents, whether the data is pipe delimited text or JSON formatted text so no extra parameters are required to be passed in by the user to process either the older or newer formatted files. Once the tool determines the format, the data is parsed into individual records. Below is an example of the parsed data in a report format.



```
License #1d38e7cc14a3070 is authenticated for business use and registered to TZWorks LLC
run time: 01/15/2019 02:55:37 [UTC]; Host: DESKTOP-M8BOQQP; MachineInfo: Windows;10.0.workstation;10.0.1;64
"cmdline: bs64 -file e:\testcase\BackstageInAppNavCache\json.format\win10\857C4EBC998F00DD83618B011A0ACC598899439A6C7B51A70C974A38EB468C48.j

LastModify UTC (entry)  | type   | DisplayName                    | Url                                                       | Author | Resourc
11/26/2018 15:28:29.938 | Folder | Custom Office Templates        | C:\Users\tzworks\Documents\Custom Office Templates        |        |
11/26/2018 00:26:49.115 | Folder | My SlickEdit Config            | C:\Users\tzworks\Documents\My SlickEdit Config            |        |
11/26/2018 02:56:13.302 | Folder | Snagit                         | C:\Users\tzworks\Documents\Snagit                         |        |
11/26/2018 00:20:47.792 | Folder | SweetScape                     | C:\Users\tzworks\Documents\SweetScape                     |        |
11/25/2018 21:22:22.513 | Folder | Visual Studio 2005             | C:\Users\tzworks\Documents\Visual Studio 2005             |        |
11/25/2018 21:01:45.173 | Folder | Visual Studio 2008             | C:\Users\tzworks\Documents\Visual Studio 2008             |        |
12/28/2018 02:09:10.808 | Folder | Visual Studio 2010             | C:\Users\tzworks\Documents\Visual Studio 2010             |        |
12/01/2018 17:32:59.600 | Folder | Visual Studio 2017             | C:\Users\tzworks\Documents\Visual Studio 2017             |        |
11/26/2018 00:52:12.312 | File   | Device Credential Guard.oxps   | C:\Users\tzworks\Documents\Device Credential Guard.oxps   |        |
11/26/2018 00:56:02.993 | File   | device guard.pdf               | C:\Users\tzworks\Documents\device guard.pdf               |        |
12/02/2018 19:09:18.672 | File   | offset                         | C:\Users\tzworks\Documents\offset                         |        |
```

Shown above is the default output, which uses a CSV variant that separates fields with pipe delimiters. One can specify other delimiters as well, by using the ***-csv_separator <delimiter arg>***. The delimiter argument can either be one of 3 characters: *comma*, *pipe* or *tab*. One can also output the parsed data in the common *Log2Timeline* format (***-csvl2t***), if desiring to merge this artifact data with other artifacts to create a timeline of activities.

If desiring to parse many files contained in a folder, one can use the ***-pipe*** option to process an entire folder and child subfolders in one session. Since the tool can sense between the different artifact internal formats, one can mix and match older and newer formatted files during this process. An example on how to use the ***-pipe*** option is shown in the command-line menu output subtitled "Usage Examples" that is shown above.

If one cannot use the ***-pipe*** option, one can use the experimental ***-enumdir*** option, which has similar functionality with more control. The ***-enumdir*** option takes as its parameter the folder to start with. It also allows one to specify the number of subdirectories to evaluate using the ***-num_subdirs <#>*** sub-option.

The last option worth mentioning is an experimental option that computes the name of the artifact file by using certain contents of the Backstage file. One invokes this via the ***-compute_name*** switch. This can be used with either the ***-file <arg>*** option or the ***-pipe*** option. This is useful to see if the contents (or the name) of the file have changed either via corruption or if done intentionally.

4 Available Options

Option	Description
-file	Specifies which Backstage file to act on. The format is: ***-file <backstage file to parse>***
-csv	Outputs the data fields delimited by commas. Since filenames can have commas, to ensure the fields are uniquely separated, any commas in the filenames get converted to spaces.
-csvl2t	Outputs the data fields in accordance with the log2timeline format.
-bodyfile	Outputs the data fields in accordance with the 'body-file' version3 specified in the SleuthKit. The date/timestamp outputted to the body-file is in terms of UTC. So if using the body-file in conjunction with the mactime.pl utility, one needs to set the environment variable TZ=UTC.
-pipe	Used to pipe files into the tool via STDIN (standard input). Each file passed in is parsed in sequence.
-enumdir	Experimental. Used to process files within a folder and/or subfolders. Each file is parsed in sequence. The syntax is ***-enumdir <folder> -num_subdirs <#>***.
-filter	Filters data passed in via STDIN via the ***-pipe*** or ***-enumdir*** options. The syntax is *-filter <"*.ext | *partialname* | ...">*. The wildcard character '*' is restricted to either before the name or after the name.
-no_whitespace	Used in conjunction with *-csv* option to remove any whitespace between the field value and the CSV separator.
-csv_separator	Used in conjunction with the *-csv* option to change the CSV separator from the default comma to something else. Syntax is *-csv_separator "|"* to change

	the CSV separator to the pipe character. To use the tab as a separator, one can use the *-csv_separator "tab"* OR *-csv_separator "\t"* options.
-dateformat	Output the date using the specified format. Default behavior is *-dateformat "yyyy-mm-dd"*. Using this option allows one to adjust the format to mm/dd/yy, dd/mm/yy, etc. The restriction with this option is the forward slash (/) or dash (-) symbol needs to separate month, day and year and the month is in digit (1-12) form versus abbreviated name form.
-timeformat	Output the time using the specified format. Default behavior is ***-timeformat "hh:mm:ss.xxx"*** One can adjust the format to microseconds, via ***"hh:mm:ss.xxxxxx"*** or nanoseconds, via ***"hh:mm:ss.xxxxxxxxx"***, or no fractional seconds, via ***"hh:mm:ss"***. The restrictions with this option is a colon (:) symbol needs to separate hours, minutes and seconds, a period (.) symbol needs to separate the seconds and fractional seconds, and the repeating symbol 'x' is used to represent number of fractional seconds.
-quiet	Show no progress during the parsing operation.
-compute_name	Computes the name of the Backstage file by looking at the file internals
-utf8_bom	All output is in Unicode UTF-8 format. If desired, one can prefix an UTF-8 *byte order mark* to the output using this option.

5 Authentication and the License File

This tool has authentication built into the binary. The primary authentication mechanism is the digital X509 code signing certificate embedded into the binary (Windows and macOS).

The other mechanism is the runtime authentication, which applies to all the versions of the tools (Windows, Linux and macOS). The runtime authentication ensures that the tool has a valid license. The license needs to be in the same directory of the tool for it to authenticate. Furthermore, any modification to the license, either to its name or contents, will invalidate the license.

6 References

1. G-C Partners Daily Blog #510, 18 Oct 2018. Office 2016 Backstage Artifacts.
2. https://blogs.technet.microsoft.com/office2010/2009/07/15/microsoft-office-backstage-part-1-backstory/
3. https://blogs.technet.microsoft.com/office2010/2009/08/11/microsoft-office-backstage-part-3-the-info-tab/