Forensic Science International: Digital Investigation 36 (2021) 301134

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

Forensic exploration on windows File History

Jisung Choi, Jungheum Park*, Sangjin Lee [1]

School of Cybersecurity, Korea University, 145 Anam-ro, Seongbuk-Gu, Seoul, South Korea

ARTICLE INFO

Article history:
Received 17 June 2020
Received in revised form
29 December 2020
Accepted 28 January 2021
Available online 21 February 2021

Keywords:
Windows forensics
File backup
File history
Forensic artifact
Forensic procedure
Multi-source data analysis
Log analysis
User behavior analysis
Anti-forensic implication
Open-source tool

ABSTRACT

Nowadays, a proliferation of flash-memory-based storage devices makes it more difficult to recover deleted files in unallocated areas. Thus, it becomes more important for forensic examiners to find and utilize backed up data generated by specially prepared *backup* features. As an interesting example of them, File History (FH) included since Windows 8 is a backup feature that can be set and operated by a user. To enable FH, it is required to select a storage device for file backup operations which can be almost any type of storage devices, including a local drive, USB flash drive, network drive, etc. This special backup feature of course allows users to restore backed up files and delete old backup versions whenever they want. Therefore, it is necessary to be able to analyze forensic artifacts that show user behaviors relating to FH, during examination of Windows systems. In this paper, we deeply explore Windows FH feature from a digital forensics perspective. As a result, this paper proposes a three-step examination procedure along with detailed considerations for each step. We also analyze impacts of several anti-forensic actions that users can perform intentionally or unintentionally. Finally, this work develops an open-source tool for identifying FH related artifacts and analyzing user behaviors on backup operations.

1. Introduction

In digital forensics and incident response (DFIR), general analysis approaches focusing on individual artifacts have limitations in analyzing usage (or modification) history of a file being used by a user. Moreover, a proliferation of flash-memory-based storage devices makes it more difficult to recover deleted files in unallocated areas. Therefore, it becomes more important for forensic examiners to find and utilize backed up data generated by specially prepared *backup* features. In that regard, modern operating systems have backup features to store *system* files in a stable state for providing against unintended errors. For example, Windows has provided special backup features such as 'restore point' and 'volume shadow copy service' to create copies of configured files or parts of volumes (Harms (2006); Hargreaves et al. (2008); Kobayaashi and Suzuki (2018)). They operate mainly when operating system's configurations are changed, but are not intended to protect *user* files.

1.1. Motivation

File History (FH), which has been included since Windows 8, is a backup feature that can be set and operated by a user. FH can be enabled with any type of storage devices, and it operates periodically according to configurations (such as 'backup cycle' and 'backup target folders') set by the user. After FH is enabled, it begins to back up files located in backup target folders. For backed up files, FH checks USN (Update Sequence Number) changes of backed up files to find modified files. Then, it only performs backup operations for modified files or newly added files to use its backup storage efficiently.

FH's operating mechanism allows users to intervene in the processes of storing files to a backup storage device and managing the files. More specifically, users can easily execute operations to delete as well as restore backed up files whenever they need. Due to this characteristic, it is necessary to be able to analyze artifacts that show user behaviors and to extract a list of backup files, during examination of Windows systems having FH feature.

During our initial steps for this work, we considered four research questions as follows:

- Where are FH related artifacts located?

* Corresponding author.
E-mail addresses: chjs207@korea.ac.kr (J. Choi), jungheumpark@korea.ac.kr (J. Park), sangjin@korea.ac.kr (S. Lee).
[1] http://forensic.korea.ac.kr

https://doi.org/10.1016/j.fsidi.2021.301134
2666-2817/

- How is it possible to correlate FH artifacts from multiple different storage devices?
- Does FH act differently in different versions of Windows?
- What kind of anti-forensic actions can be performed and what are their impacts on forensic investigations?

1.2. Contributions

This paper explores Windows FH feature for digital forensic activities. We divide FH forensics into three steps and explain detailed considerations for each step. The steps are as follows: [step-1] identifying FH traces inside target storage devices, [step-2] determining all devices relating to FH, and [step-3] extracting backup operation history and analyzing actual backed up files. This paper then analyzes impacts of several anti-forensic actions that users can perform while using FH feature. The actions are as follows: [action-1] deleting backed up file(s), [action-2] hiding file(s) through *restore* operation, [action-3] disabling and re-enabling FH, and [action-4] operating FH while a backup storage is not working. Finally, we develop an open-source tool (parsers and plugins for plaso (2019)) to interpret FH related artifacts for cybersecurity community. In summary, this paper makes the following contributions:

- This work proposes a detailed three-step examination procedure for FH related artifacts.
- This work analyzes impacts of four potential anti-forensic actions associated with FH.
- This work demonstrates that the proposed results are useful and informative, by developing an automated tool.

The remainder of this paper is as follows: Section 2 reviews previous works on FH. Section 3 describes FH artifacts that need to be checked during DFIR. Section 4 analyzes impacts of several anti-forensic actions. Section 5 introduces a tool developed to analyze FH artifacts. Finally, Section 6 concludes this work and discusses future directions.

2. Background

2.1. Previous works and existing tools on File History

Johnson (2013) and Khizra and Quba (2014) studied several forensic artifacts of FH in Windows 8. Both studies focused on Config.xml which is a configuration file having various fields such as user name, PC name, backup target folders, and backup storage volume name. They also identified the last backup time recorded in a specific key to Windows registry, and briefly described event log entries relating to FH. However, they did not deal with actual backed up files and Catalog.edb (EDB: Extensible Storage Engine Database) that manages a detailed list of them. Dehaviland (2017) also briefly described several Windows registry keys created by FH operations.

Windows 10 provides a special feature, Timeline, which records user behaviors that can be useful in event analysis. Horsman et al. (2019) studied various data stored in ActivitiesCache.db where Timeline events are recorded, and then they proposed methods to extract meaningful information from the database. For forensic activities, the Timeline database can be used for the same purpose as FH in that it allows examiners to identify a history of file operations. Since Timeline only stores operations on a file, better understanding of the file related history may be obtained by analyzing it together with metadata and content of the file stored by FH.

During literature analysis, we have identified that two free tools, Autopsy and plaso, provide abilities (as a form of plugin) to interpret and extract information from Catalog.edb files of FH (McKinnon (2017); plaso (2019)). First, a plugin of Autopsy can process a single Catalog.edb file and extract several information including file name, size, USN, created/modified/backup timestamps, etc. However, the current version has a limitation that it does not process Catalog.edb files stored in other locations like backup storage devices. Second, a plugin of plaso utilizes 'namespace' and 'string' tables to extract several informative fields including ID, parent ID, USN, attribute and name of each file. It also can extract created and modified timestamps relating to each file from 'namespace' table. As we all know, although plaso is able to identify and analyze multiple Catalog.edb files regardless of their paths in an input image (or device), the current version of the plugin has some limitations as follows: (a) it is not possible to know when target files were backed up, (b) it does not support to reconstruct full paths of backed up file, and (c) it does misinterpret relationships between 'namespace' and 'string' tables while getting file/folder names.

In addition, both tools have the following common limitations: (a) they do not process several informative values in Config.xml, which has configurations for FH like backup cycle and backup storage information, and (b) they do not handle FH related information stored in other Windows artifacts.

As far as we know, as of Dec 2020, there are no other commercial and open-source tools that support forensic features on FH related artifacts, except for the above-mentioned tools and our own tool developed by this work.

2.2. Research direction based on literature review

Existing materials on FH are mostly web-posts or presentations like technical memos aimed at introducing it to the forensics community, and the only research paper in which it was mentioned just briefly explains a small part of it.

Therefore, this paper's objective is to build a deeper and broader understanding of FH operations by carrying out systematic experiments based on the basic information introduced by previous works, in order to support practical digital forensic activities. In other words, this study's direction is to improve the theoretical knowledge and practical forensic techniques related to FH, through discovering new characteristics that all the previous literature did not mention as well as overcoming limitations of existing tools. Furthermore, since the findings in the literature were based mainly on Windows 8, FH artifacts need to be updated with different versions of Windows 10. For reference, our experiments target Windows 10 version 1909 or earlier.

3. File History related forensic artifacts

As mentioned above, FH is not a feature enabled by default. However, if a target image (having a Windows system partition) has usage traces of FH, meaningful information can be obtained from various artifacts in the target itself as well as related backup storage devices. For a thorough analysis, this section begins with a proposal of a systematic examination procedure to obtain FH related artifacts. For reference, Appendix Table A1 provides an integrated view of the artifacts found in this study.

3.1. A three-step procedure for File History examination

The proposed procedure consists of: [step-1] identifying FH traces inside target storage devices, [step-2] determining all devices relating to FH, and [step-3] extracting backup file information and analyzing actual backed up files in backup storage devices. As

shown in Fig. 1, an examiner needs to perform the second and third steps, if there exist interesting traces found during the first step.

The following sub-sections describe details of each step for analyzing FH related forensic artifacts.

3.2. Identifying File History usage traces

When FH is enabled, configuration XML files (Config1.xml and Config2.xml, which are generally identical in a host system, so from here they are called collectively 'Config.xml') are created at specific folders on both a host Windows system and a backup storage device, as shown in Appendix Table A1.

A configuration XML file contains backup options including 'backup cycle' and 'backup retention period'. While configuring FH, a user can select an appropriate backup cycle among the following options: 10 min, 15 min, 20 min, 30 min, 1 h (default), 3 h, 6 h, 12 h and 24 h. And then, the selected value is recorded in seconds to 'DPFrequency' element of a configuration XML file. There are also several selectable options on the backup retention period that is the amount of time that a backup is held before being purged. Table 1 lists those options and associated values recorded in 'RetentionPolicyType' and 'MinimumRetentionAge' elements of a configuration XML file. Moreover, it is possible to know when backup options of FH were initially configured or updated, through identifying the creation and modification timestamps of configuration XML files.

In addition, as shown in Appendix Table A1, FH creates the following two registry artifacts: *Start* value of *fhsvc* key and *ProtectedUpToTime* value of *FileHistory* key. If FH is enabled, the startup type of *fhsvc* (File History Service) will be 'automatic' (integer 2), but if not it will be 'manual' (integer 3). That is, an integer is assigned to *Start* value in *fhsvc* key path for configuring the current startup setting. Next, *ProtectedUpToTime* value of *FileHistory* key path has the last backup execution time in Windows 64-bit time format.

Furthermore, FH creates Windows event records on periodic backup operations with the two dedicated EVTX (Windows XML Event Log format) files. The WHC.evtx is used for managing events relating to backup starts and ends. The start and end of a backup operation can be separated by *hc_stateid* value of an event data: i.e., 1023 and 767 means 'start' and 'end' respectively. In addition to that, the BackupLog.evtx exists for 'warning' and 'error' events that can occur during backup operations.

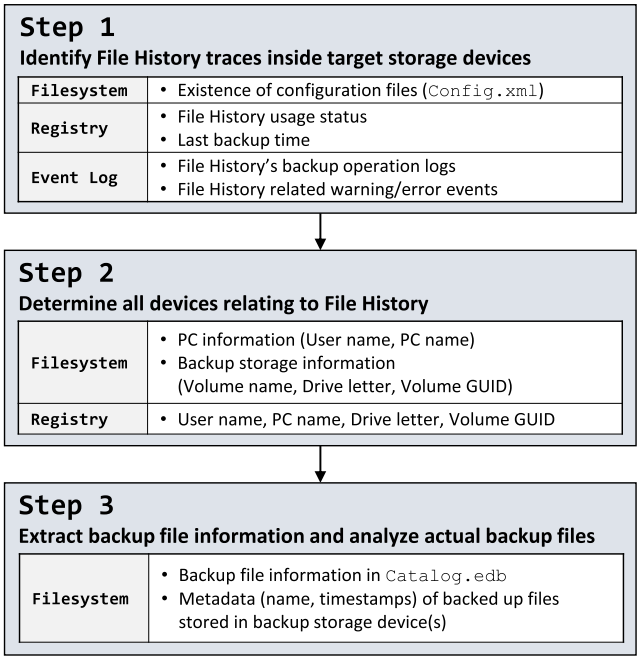


Fig. 1. A three-step procedure for File History examination.

3.3. Determining all associated storage devices

Along with backup options, Config.xml manages user and device related information. They include specific identifiers associated with, (1) a host machine having files that will be backed up, (2) a user account who enabled FH, and (3) a storage device dedicated for file backup. Thus, through analyzing configuration XML files, examiners can identify host Windows systems as well as storage devices used for FH activities.

Regarding FH forensics, it is generally simple to show a connection between a host Windows system and an active associated backup storage, if identical configurations are identified from both of them. In addition, even if Config.xml exists only on one side of them, there still exist ways to find a connection through identification values from the registry of host Windows systems. Several well-recognized registry items such as RegisteredOwner, ComputerName, and MountedDevices are useful for doing that.

As shown in examples of Fig. 2, Config.xml has a target (i.e., backup storage) element including volume name, volume GUID (Globally Unique Identifier), drive letter, and drive type. Especially, the volume GUID can be identified from the registry in a host Windows system as well as a backup storage device. That is, values in MountedDevices key contain identifiers of mounted storage devices that include a disk signature in MBR (Master Boot Record) and partition GUIDs in GPT (GUID Partition Table). Therefore, it is possible to find a storage device for backup relating to a FH enabled host, as shown in Fig. 2. For reference, the drive type can be one of 'Fixed', 'Removable', and 'Remote' according to the target's connection type.

3.4. Extracting backed up file information and analyzing metadata of backed up files

3.4.1. Extracting backed up file information

After identifying FH related host Windows systems and storage devices from Config.xml files, it is necessary to extract a list of backed up files and analyze metadata of the files using EDB (Extensible Storage Engine database) files located in the above-mentioned configuration folders. There are two EDB files, Catalog1.edb and Catalog2.edb, used for managing backup operations. Like Config1.xml and Config2.xml, the two EDB files are generally identical in a host Windows system. However, in case of a backup storage device, they interestingly take turns storing metadata for every backup operation. That is, because the two EDB files are updated one by one, there is a chance to overlook recently updated (backed up or deleted) files if examiners focus on just one EDB file. From here they are called collectively 'Catalog.edb' for simple representation.

Catalog.edb consists of four primary tables including 'backupset', 'file', 'namespace', and 'string'. As shown in Table 2, there exist useful artifacts such as backed up files's full paths, their original created/modified timestamps, and logs on backup operations. The *id* of 'string' table is associated with the *childId* and *parentId* of 'file' as well as 'namespace' tables. The *id* of 'file' table is also linked to the *fileRecordId* of 'namespace' table. In addition, the *tCreated* of 'namespace' table and the *id* of 'backupset' table are used to join the two tables together.

Fig. 3 shows an example of extracting information on a backed up file, test10.txt, from various tables in Catalog.edb. In this

Table 1
RetentionPolicyType and MinimumRetentionAge values according to the selected backup retention period option.

Selectable option	RetentionPolicyType	MinimumRetentionAge
Until space is needed	NO LIMIT	365
1 month	AGE LIMIT	1
3 month	AGE LIMIT	3
6 month	AGE LIMIT	6
9 month	AGE LIMIT	9
1 year	AGE LIMIT	12
2 year	AGE LIMIT	24
Forever	DISABLED	the previously set value is retained

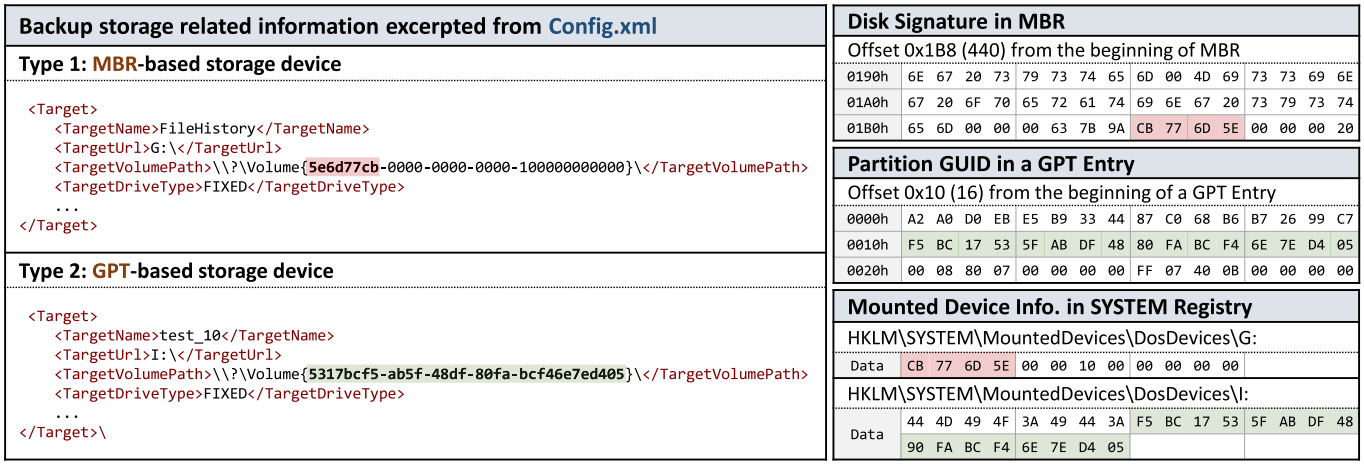


Fig. 2. Determination of backup storage devices relating to File History using identifiers recorded in Config.xml, registry, MBR and GPT.

Table 2
Tables and columns in Catalog.edb

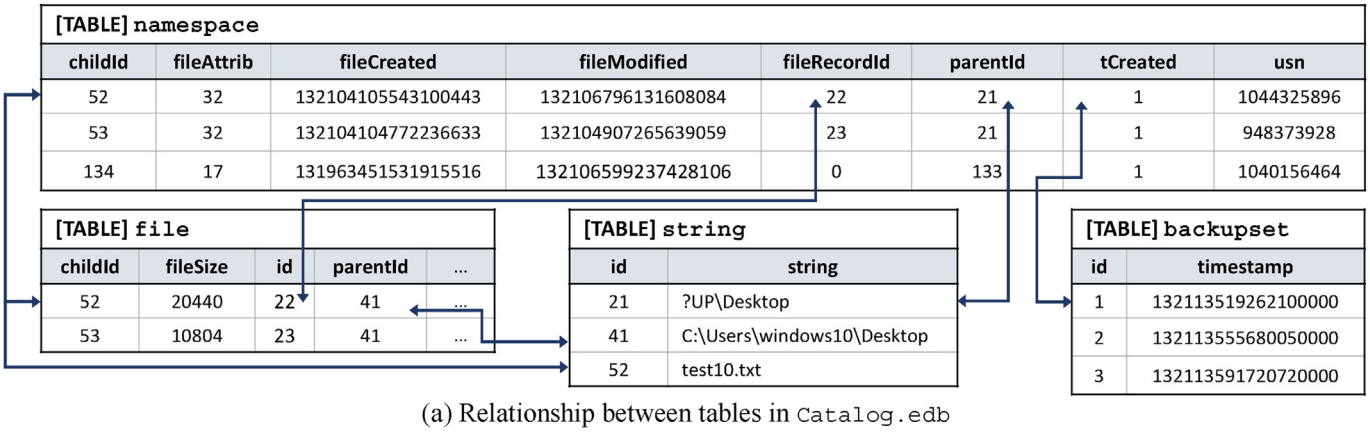
Table	Column	Description (all time-related values are Windows 64-bit timestamps)
backupset	id	Index of backup operation
	timestamp	Start time of backup operation
file	id	Index of file
	childId	Foreign key to *id* column of 'string' table (for a file's name)
	parentId	Foreign key to *id* column of 'string' table (for a folder where *childId* is located)
	fileSize	Backed up file's size
namespace	childId	Foreign key to *id* column of 'string' table (for a file or folder's name)
	parentId	Foreign key to *id* column of 'string' table (for a folder where *childId* is located)
	fileRecordId	Foreing key to *id* column of 'file' table
	fileCreated	File creation time
	fileModified	File modification time
	tCreated	Foreign key to *id* column of 'backupset' table
	usn	Backed up file's USN
string	id	Index of string
	string	Backed up file's name or folder path

example, *id* of the file's name (test10.txt) in 'string' table is 52, and the value is associated with a record having *childId* 52 in 'file' table. In this manner, as *parenId* 41 of the file record is linked to *id* 41 of 'string' table, the backed up file's size is 20,440 bytes and its original location is `C:\Users\windows10\Desktop\`. Moreover, through id 22 in the file record, it is possible to associate it with a record having fileRecordId 22 in 'namespace' table. As shown in the figure, the namespace record contains two timestamp columns: fileCreated (132104105543100443) and fileModified (132106796131608084). In addition, because tCreated 1 of 'namespace' table is associated with id 1 of 'backupset' table, we can identify the start time of a backup operation (132113519262100000). Consequently, test10.txt file was originally located at `C:\Users\windows10\Desktop\test10.txt`, and its created and modified timestamps are 2019/08/16 06:29:14 (UTC) and 2019/08/19 09:13:33 (UTC), respectively. Also, the file was backed up on 2019/08/27 03:58:46 (UTC).

It should be noted here that 'file' table does not contain folder related information, so *fileRecordId* column in 'namespace' table is always assigned to 0 for folders.

3.4.2. Analyzing metadata of backed up files

After extracting a list of backed up files, examiners can dig into actual backed up files, which can be stored at either a host Windows system or a backup storage device depending on FH's operational situations. They can be stored in the 'Data' folders as shown



[TABLE] namespace

childId	fileAttrib	fileCreated	fileModified	fileRecordId	parentId	tCreated	usn
52	32	132104105543100443	132106796131608084	22	21	1	1044325896
53	32	132104104772236633	132104907265639059	23	21	1	948373928
134	17	131963451531915516	132106599237428106	0	133	1	1040156464

[TABLE] file

childId	fileSize	id	parentId	...
52	20440	22	41	...
53	10804	23	41	...

[TABLE] string

id	string
21	?UP\Desktop
41	C:\Users\windows10\Desktop
52	test10.txt

[TABLE] backupset

id	timestamp
1	132113519262100000
2	132113555680050000
3	132113591720720000

(a) Relationship between tables in Catalog.edb

Original 'test10.txt' File	
Filesystem Metadata	
Name	test10.txt
DirectoryName	C:\Users\windows10\Desktop
Directory	C:\Users\windows10\Desktop
IsReadOnly	False
Exists	True
FullName	C:\Users\windows10\Desktop\test10.txt
Extension	.txt
CreationTime	2019/08/16 06:29:14 (UTC)
LastAccessTime	2019/11/01 09:51:16 (UTC)
LastWriteTime	2019/08/19 09:13:33 (UTC)
Attributes	Archive

Backed up 'test10.txt' File	
Filesystem Metadata	
Name	test10 (2019_08_27 03_58_46 UTC).txt
DirectoryName	K:\FileHistory\windows10\DESKTOP-MT2660C (2)\Data\C\Users\windows10\Desktop
Directory	K:\FileHistory\windows10\DESKTOP-MT2660C (2)\Data\C\Users\windows10\Desktop
IsReadOnly	True
Exists	True
FullName	K:\FileHistory\windows10\DESKTOP-MT2660C (2)\Data\C\Users\windows10\Desktop\test10 (2019_08_27 03_58_46 UTC).txt
Extension	.txt
CreationTime	2019/08/27 03:58:48 (UTC)
LastAccessTime	2019/11/01 04:14:23 (UTC)
LastWriteTime	2019/08/19 09:13:33 (UTC)
Attributes	ReadOnly, Archive

(b) Comparison between an original file on C drive and its copied file on K drive

Fig. 3. Extracting backed up file information in Catalog.edb

in Appendix Table A1.

FH basically saves files in the 'Data' folder of a backup storage device. Each backed up file maintains its original folder structure and modification timestamp, but creation and access timestamps on the file are newly assigned when it is created, just like 'copy' operation. We experimentally confirmed that the characteristics are the same regardless of what filesystems (e.g., FAT32, exFAT, NTFS, and ReFS) are in use for backup storage devices. In addition, each backed up file will have a specially-formatted path 'PATH\NAME (TIME).EXT' as shown in Fig. 3-(b): PATH is the original folder path; NAME is the original file name; TIME is a sting timestamp in UTC (Coordinated Universal Time) that means the time of a backup operation; and EXT presents the original extension of the backed up file. Thus, if a previously backed up file is modified, a new backup operation will create a new file at the same PATH, and the name of the newly backed up file will be composed of its original NAME, a *new* TIME and its original EXT.

The 'Data' folder managed by backup operations may contain a folder named $OF, which is used for files that have paths longer than 146 characters. As shown in Fig. 4, each backed up file will have a specially-formatted path 'ID1\ID2 (TIME).EXT': ID1 and ID2 are *id* values of 'string' table which indicate a folder path and a file name respectively; TIME is a sting timestamp in UTC; and EXT means the extension of the file.

Regarding the 'Data' folder in a host Windows system, it can be used temporarily for storing files when FH does not work normally, for example a backup storage does not have enough capacity or a backup storage is disconnected. Each sub-folder of the 'Data' folder has a special name which is one of *id* values in 'backupset' table that means a unique index of each backup operation. Afterwards, temporarily stored files will be moved to a backup storage when FH gets back to normal condition.

3.5. Artifact changes between different windows versions

The above-mentioned artifacts differ depending on Windows versions. In Windows 8 and 8.1, the default backup folders of FH are as follows: Documents, Music, Pictures, Videos, Contacts, Desktop, and Favorites. In addition to that, Windows 10 added the following folders: Pictures\Saved Pictures, Pictures\Camera Roll, Searches, 3D Objects, OneDrive, Saved Games, Links, and Downloads. Windows 10 also allows users to add any folders for backup. It should be noted that a list of backup target folders can be found in 'User-Folder' element in Config.xml in all versions of Windows.

For reference, in Windows 10 version 1709 or earlier, FH allows users to use a network location shared through HomeGroup feature, which is a group of Windows systems connected to the same local area network. If a network location is configured as a backup storage through HomeGroup, a specific registry key (see Appendix Table A1) stores related information.

The HomeGroup related registry key consists of five registry values as follows: *DeviceType*, *FolderPath*, *FriendlyName*, *ShareName*, and *Url*. First, the *DeviceType* has '2' if the current user created HomeGroup for sharing, or '1' if the current user joined someone else's HomeGroup. Second, the *FolderPath* normally records the full path of a backup location for FH operations, but it can only have a drive letter if the current user simply joined someone else's HomeGroup. Third, the *FriendlyName* has the volume name of a FH backup storage. Fourth, the *ShareName* stores the name of a FH backup storage shared through HomeGroup, but this value does not

[TABLE] string

id	string
126	C:\Users\jisung\Favorites\timestamp
127	SANS Digital Forensics and Incident Response Blog How to Preserve Cyber Investigation Evidence Screencast Tool SANS Insti.url

(a) File and folder names in 'string' table

'127 (2019_04_17 02_44_17 UTC).url' file inside $OF folder

Filesystem Metadata	
Name	127 (2019_04_17 02_44_17 UTC).url
DirectoryName	F:\FileHistory\jisung\DESKTOP-JISUNG (15)\Data\$OF\126
Directory	F:\FileHistory\jisung\DESKTOP-JISUNG (15)\Data\$OF\126
IsReadOnly	True
Exists	True
FullName	F:\FileHistory\jisung\DESKTOP-JISUNG (15)\Data\$OF\126\127 (2019_04_17 02_44_17 UTC).url
Extension	.url
CreationTime	2019/04/17 02:45:07 (UTC)
LastAccessTime	2019/04/17 02:45:07 (UTC)
LastWriteTime	2019/04/17 00:57:02 (UTC)
Attributes	ReadOnly, Archive

(b) Metadata of an example file stored in $OF folder

Fig. 4. Example of identifying file and folder names in the $OF folder using 'string' table in Catalog.edb

have any data if the current user joined someone else's HomeGroup. Finally, the *Url* records the URL (i.e., \?PCName?\?ShareName?) of a shared network drive for FH in HomeGroup.

It should be note that although HomeGroup feature removed from Windows 10 version 1803 (Microsoft (2018)), users can still configure a shared network drive as a backup storage for FH.

4. Impact analysis of anti-forensics on File History

FH allows users to delete old backup versions and restore backed up files whenever they want. These features are essential for backup operations, but they also can be used to perform anti-forensics intentionally or unintentionally. Thus, this section analyzes impacts of four potential anti-forensic actions, in order to support examiners who need to deeply understand any abnormal information that could be created as FH related artifacts.

4.1. Deleting backed up files

FH provides a special feature, 'Clean Up Versions' (CUV), that allows users to delete previous backups. This feature needs to be executed with an option for deleting a specific backup period. All selectable backup periods for CUV include *All but the latest one*, *Older than* 1 month, *Older than* 3 months, *Older than* 6 months, *Older than* 9 months, *Older than 1 year* (default), and *Older than 2 years*. Note that if a backed up file is not modified since its first backup, CUV does not delete the file even if it corresponds to the period selected by a user.

Fig. 5 shows how CUV works. The A'.doc is an additional file backed up by modifying A.doc. As shown in the Figure, CUV deletes A.doc, which was backed up before 'T2' (Order than 1 year) as a user-selected backup period. However, the B.doc backed up prior to 'T0' will not be deleted because there does not exist additional versions of the file.

When CUV deletes a specific backup file, its associated information are also deleted from 'namespace' table in Catalog.edb. Note that our experiments showed that it is not a secure deletion (wiping), so there is a possibility that deleted records will be recovered from unused areas in FH related EDB files. In addition, from a filesystem perspective, examiners can recover metadata as well as contents from unallocated areas only if they are not overwritten. In particular, in the case of NTFS filesystem, a backed up file's *created* (backup time), *accessed* (backup time) and *modified* (modified time of the original file of it) timestamps do not change as a result of a CUV operation, and only *entry modified* time changes to the time when it was deleted. In other words, if a backup storage use FAT (or exFAT) filesystem, a backed up file's all three (created, accessed and modified) timestamps remain unchanged when the file was deleted by CUV. Thus, above findings mean that examiners are likely to find additional information about FH's operations through data recovery.

4.2. Hiding files through restore operation

The 'Previous Versions' (PV) feature provided in the 'Properties' of Windows Explorer shows a list of files backed up by FH. The PV allows users to selectively *open* or *restore* a file backed up at a certain point in time. An interesting point is that PV determines the relationship between a file and its backup files, by checking only the full paths of them.

Fig. 6-(a) shows PV of a text file filled with 'Restore' in C:\Users\windows10\Desktop\, and the there exists a backup version of the file with a modified time of 2019/11/21 15:21. In addition, Fig. 6-(b) indicates a newly created file filled with 'Fake Restore' in the same location after deleting the text file. As a result, checking PVs of both the old and new files demonstrates that they have an identical backed up file. These characteristics of FH allow users to hide a specific file by the following steps:

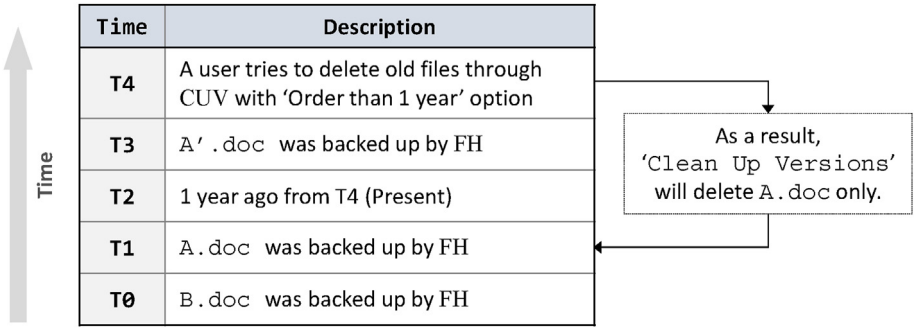
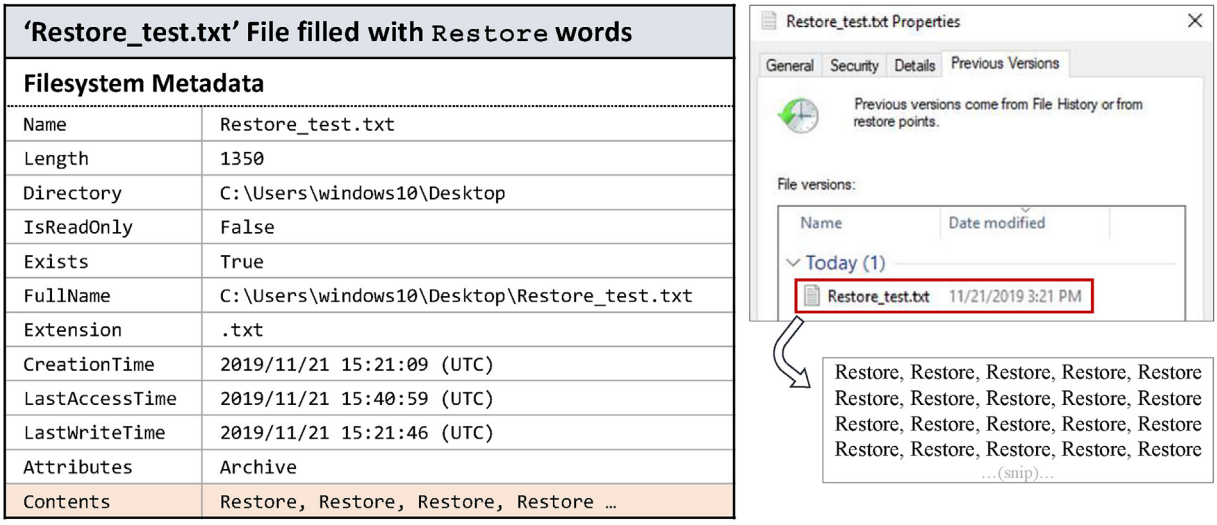
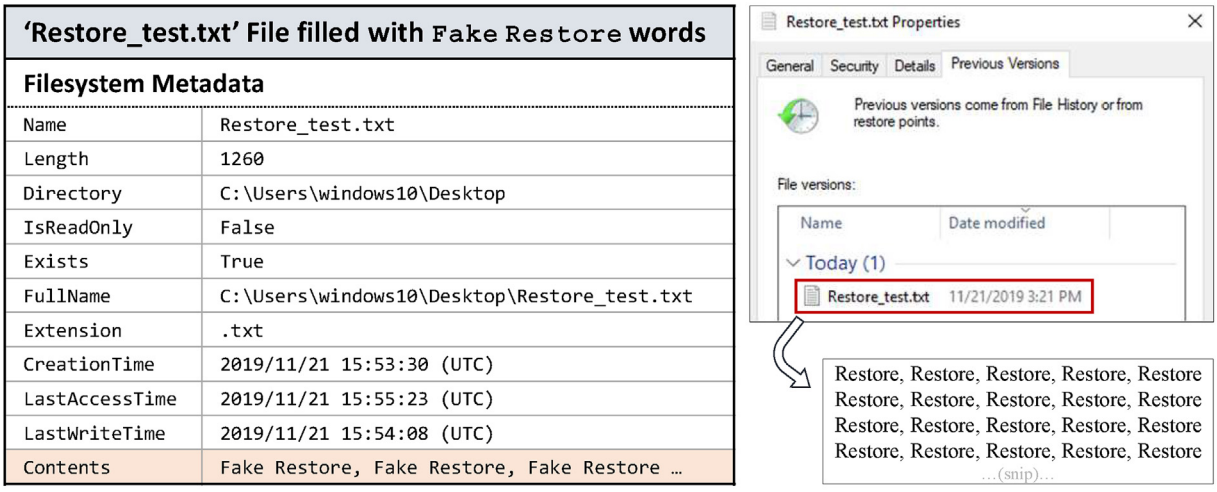


Fig. 5. An example operation of Clean Up Versions.



(a) An example TXT file which was backed up on 2019/11/21 15:21



(b) An example 'fake' TXT file with the same name and extension of a previous file

Fig. 6. An example operation of Previous Versions: path-based linkage.

1. Select a file F_a that has at least a previously backed up version.
2. Execute FH to back up the current version the selected file F_a.
3. After deleting the file F_a, create a new file F_b with the same name.
4. Wait for FH's backup operation to back up the new file F_b.
5. After completing the backup operation for F_b, restore F_a (that was backed up in the step 2) deliberately through PV's *restore* operation in order to hide F_b.

In our experiments, we found an artifact that could be used to respond to this anti-forensic activity. Specifically, when executing a

Example of `Restore.log`

	unknown	fileRecordId	Restored file	usn	fileAttrib	fileCreated	fileModified	
<	1 2	7c	C:\Users\windows10\Desktop\Restore_test.txt	53c62bf0	20	1d5a03862637749	1d5a033f3447875	>

Fig. 7. A line within an example 'Restore.log' file generated *restore* operation.

restore operation through PV, a text log file is created at `%LocalAppData%\Microsoft\Windows\FileHistory\Configuration\Restore.log`.

Fig. 7 shows an example 'Restore.log', which stores the size, path, USN, created and modified timestamps of restored files. Each line of this example log includes information about a restored file that has 0x53c62bf0 (1405496304) as its USN. Afterwards, when FH tries to back up the file again, its information ($R_{restored}$) is recorded in 'namespace' table in Catalog.edb like Fig. 8. For reference, in the figure, $R_{original}$ and $R_{restored}$ records have the same *fileRecordId* value, because an existing record of 'file' table for $R_{original}$ could be utilized for $R_{restored}$ to determine its path, name and size information. Therefore, if there exist records with the same *childId* and *parentId* values as well as with different *fileCreated* values like R_{hidden}, it is necessary to consider the possibility of intentional hiding of data.

4.3. Disabling and re-enabling of a backup storage device

In modern Windows systems, there are two approaches for configuring FH options: *Control Panel* (legacy) and *Settings* (new). The *Settings* is a Windows system app for supporting configuration settings and it is designed to be touch-friendly for modern devices. One interesting thing is that each approach offers different functions for configuring a backup storage device for FH.

First, *Settings*'s Backup menu provides the ability to disable an active backup storage device as well as add (i.e., select and enable) a storage device for FH backup through 'Add a drive' button. If a backup storage device is selected for FH, a new backup folder is created for backup operations at the following path within the backup storage:

- \FileHistory\?*UserName*?\?*PCName*?\

A user of course can reuse an existing storage device previously used for FH backup in the current Windows system. In this case, FH tries to create a new backup folder by appending *a number* to the end of an existing backup folder name. That is, as shown in Fig. 9, there can exist multiple FH backup folders using sequential numbering, if a user frequently resets FH. Therefore, examiners need to check the current (active) backup storage path through

[TABLE] `namespace`

File ID	childId	fileAttrib	fileCreated	fileModified	fileRecordId	id	parentId	...	tCreated	usn
$R_{original}$	172	32	132187908691099033	132187909062424693	124	223	21	...	109	1343760080
R_{hidden}	172	32	132187928106596169	132187928486739805	127	231	21	...	110	1390765784
$R_{restored}$	172	32	132187928106596169	132187909062424693	124	238	21	...	111	1405496304

Fig. 8. Example records of 'namespace' table relating to *restore* operation.

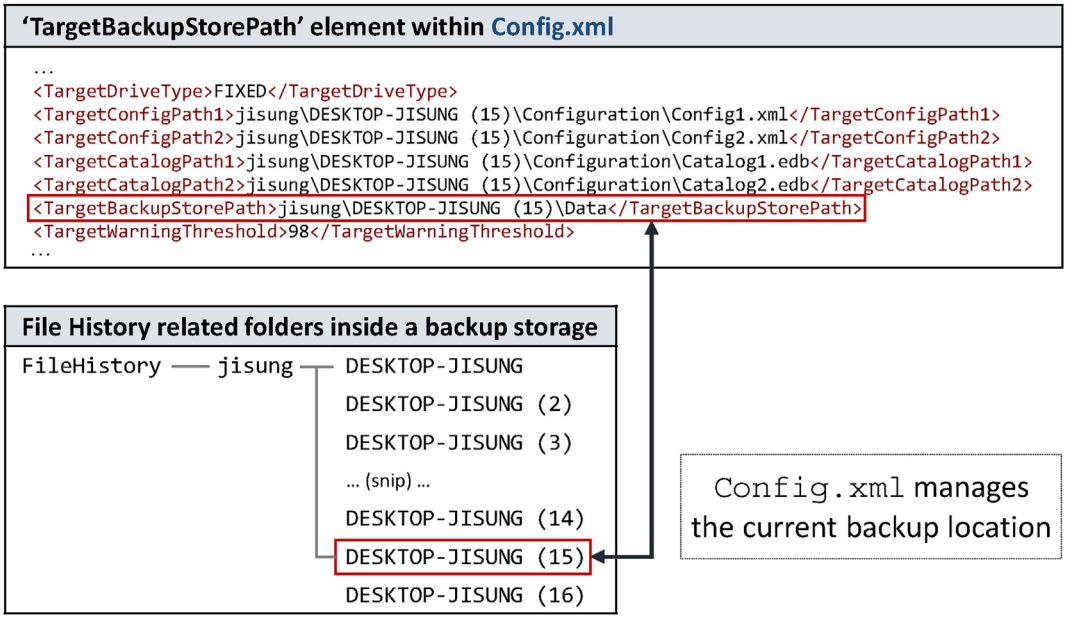


Fig. 9. An example of multiple backup folders created by re-enabling File History.

Config.xml, and also need to perform a consolidated analysis of all backed up files stored in multiple backup folders.

Second, *Control Panel*'s FH menu allows users to stop FH, but it does not provide the ability to disable an active backup storage device. Instead, users can select a new storage device for FH backup through 'Select Drive' button. Unlike a result of *Settings*, if a previously used backup folder exists on the selected storage device, FH will continue backing up files to that folder. In other words, it should be noted that contents of Config.xml and sub-folder names of an active backup folder might not match as shown in Fig. 10: e.g., *DESKTOP-MT2660C* and *windows8*.

In addition, as listed in Appendix Table A1, there is an interesting registry value, TargetChanged, which is updated to 0x00000001 (REG_DWORD) when a configured backup storage is changed through *Settings*. It should be noted that this registry value will not be updated when the backup storage is changed in *Control Panel*.

4.4. Disconnecting an active backup storage device

In FH, the ability to utilize external storage devices for backup means that an active backup storage can be easily disconnected from a running PC. In our experiments, when a backup storage device was detached during backup operations, FH began to back up target files to 'Data' folder in a host Windows system from the time of disconnection. To respond to this activity, examiners can check *error* messages (EventID 201 and EventLevel 2 - 'Unable to scan user libraries for changes and perform backup of modified files for configuration') in 'BackupLog.evtx'.

Moreover, FH does not automatically operate unless a storage device is configured and connected properly. However, our experiments revealed that backup operations can forcibly work by clicking 'Run now' button directly in FH function of *Control Panel*. In this case, an error message appears asking users to connect a valid backup storage device, but FH operates to back up files temporarily to 'Data' folder in a host Windows system. It should be noted that examiners can identify related *warning* messages (EventID 204 and EventLevel 3 - 'Unusual condition was encountered during finalization of a backup cycle for configuration') from 'BackupLog.evtx'.

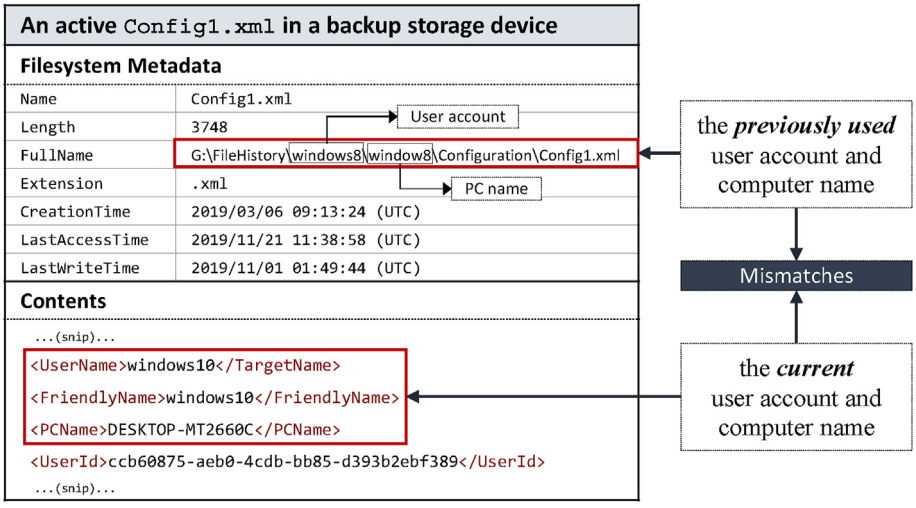


Fig. 10. An example of mismatch between contents of Config.xml and sub-folder names of an active backup folder.

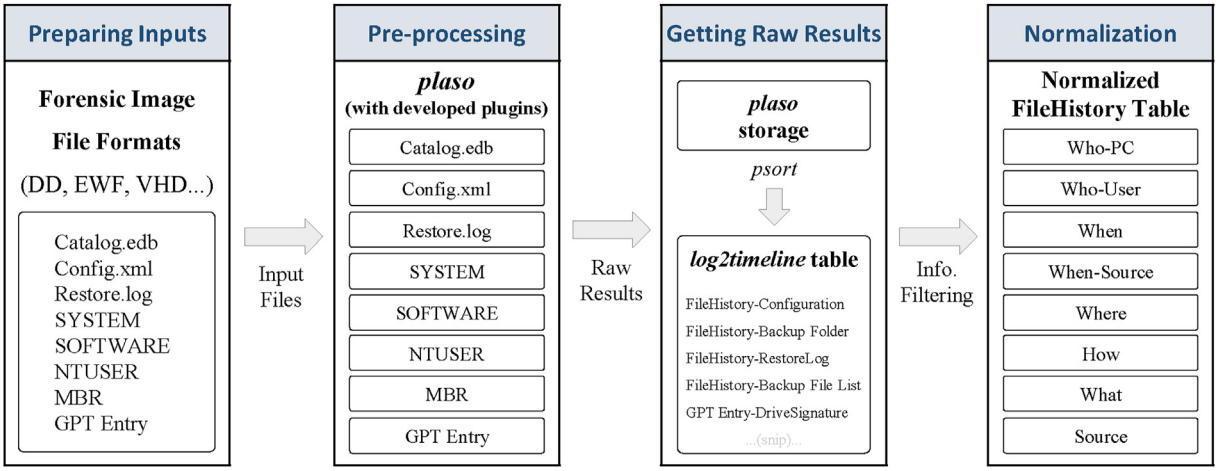


Fig. 11. Workflow of EFIC

ID	Who		When	When-Source	Where*	How	What	Source
	PC	User						
1	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Host	include folder(s)	C:\BackupIncludeFolder	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Config1.xml
2	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Host	exclude folder(s)	C:\BackupExcludeFolder	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Config1.xml
3	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Host	set 'Retention Policy'	Forever	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Config1.xml
4	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Host	set 'Backup Cycle'	3600 seconds	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Config1.xml
5	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Host	set a backup storage	test_10 (E:), {c2b3c428-7780-4b6c-a785-7a79d4c620ba}, FIXED	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Config1.xml
6	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Backup	include folder(s)	C:\BackupIncludeFolder	USBStor.E01\FileHistory\Windows10x64\DESKTOP-GMH5TJM\Configuration\Config1.xml
7	DESKTOP-GMH5TJM	Windows10x64	2020-03-09 T03:39:32Z	modified time of source	Backup	exclude folder(s)	C:\BackupExcludeFolder	USBStor.E01\FileHistory\Windows10x64\DESKTOP-GMH5TJM\Configuration\Config1.xml
8	DESKTOP-GMH5TJM	-	2020-03-09 T03:39:32Z	registry value's QWORD data	Host	execute a backup	the last backup operation	Win10.E01\Users\Windows10x64\NTUSER.DAT\[Software\Microsoft\Windows\CurrentVersion\FileHistory\ProtectedUpToTime]
9	DESKTOP-GMH5TJM	-	2020-01-20 T13:43:50Z	registry key's last written time	Host	set a backup storage through HomeGroup	test_8 (E:), \\WIN-1GIMMFRIB7B (c)	Win10.E01\Windows\System32\config\SOFTWARE\[Microsoft\Windows\CurrentVersion\FileHistory\HomeGroup\Target\]
10	DESKTOP-GMH5TJM	-	2020-01-15 T06:29:07Z	*timestamp* column of 'backupset' table	Host	back up a file	C:\Users\Windows10x64\Desktop\0000000000\1111111111.txt, [ID: 20, USN: 18606776, Size: 288]	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Catalog1.edb
11	DESKTOP-GMH5TJM	-	2020-01-15 T06:27:22Z	*fileModified* column of 'namespace' table	Host	modify a file	C:\Users\Windows10x64\Desktop\0000000000\1111111111.txt, [ID: 20, USN: 18606776, Size: 288]	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Catalog1.edb
12	DESKTOP-GMH5TJM	-	2020-01-15 T06:26:12Z	*fileCreated* column of 'namespace' table	Host	create a file	C:\Users\Windows10x64\Desktop\0000000000\1111111111.txt, [ID: 20, USN: 18606776, Size: 288]	Win10.E01\Users\Windows10x64\AppData\Local\Microsoft\Windows\FileHistory\Configuration\Catalog1.edb
13	DESKTOP-GMH5TJM	-	2020-01-15 T06:18:36	registry key's last written time	Host	use a storage device	(E:), {c2b3c428-7780-4b6c-a785-7a79d4c620ba}	Win10.E01\Windows\System32\config\SYSTEM\[MountedDevices\]
14	-	-	-	-	Backup	use a storage device	{c2b3c428-7780-4b6c-a785-7a79d4c620ba}	USBStor.E01\[GPT's Partition Entry Array]

(a)

(b)

* [Host]: a host Windows system's drive, [Backup]: a backup storage device.
(a) Identical configurations can be found in `Config1.xml` files stored in both a host Windows system (`WIN10.E01`) and a backup storage device (`USBStor.E01`).
(b) Identical drive letter and GUID values can be found from multiple locations including a `Config1.xml`, a `SYSTEM` hive and a `GPT` partition entry.
(c) A storage device (having volume name `test_8` and drive letter `E:`) shared by a PC named 'WIN-1GIMMFRIB7B' through HomeGroup feature was used for FH backup.

Fig. 12. An example normalization of File History artifacts from multiple sources.

5. Implementation - EFIC: extract file history IntelligenCe

5.1. Design of EFIC

EFIC extracts meaningful information from files where FH related artifacts are stored. EFIC written in Python3 consists of two phases: 'pre-processing' and 'information filtering and normalization', as shown in Fig. 11. First, 'pre-processing' is a phase to get raw artifacts based on *plaso*. This phase takes a forensic image and extracts information from FH related files described in Section 3. Since the current version of *plaso* has limited support for FH, we have improved the existing functions as well as developed new parsers/plugins to analyze the artifacts (Catalog.edb, Config.xml, Restore.log, several registry keys/values, MBR, and GPT) introduced in this paper. Second, 'information filtering and normalization' phase normalizes the extracted information and stores it in a result database. The following sub-section describes FH data normalization in detail.

For reference, the source code[2] under development is open-sourced for anyone who want to use, improve and verify.

5.2. File History related data normalization and its application

To explain data normalization, it is assumed that the following two forensic images were acquired during investigation: (1) Win10.E01 is the image of a storage device containing a Windows system partition, and (2) USBStor.E01 is the image of an external storage device which uses GPT partition tables.

As shown in an example in Fig. 12, a normalized result table produced by EFIC consists of columns including *Who* (computer and user), *When*, *When-Source*, *Where*, *How*, *What,* and *Source* (fullpath), in order to clearly represent the meaning of each record. For example, the first record from a Config.xml shows that a user ('Windows10x64') on a PC named 'DESKTOP-GMH5TJM' added 'C:\BackupIncludeFolder\' as a backup target folder of FH at least before 2020-03-09T03:39:32Z. Also, the 11th record from a Catalog.edb informs that a text file (usn: 18606776, size: 288 bytes) was backed up at 2020-01-15T06:29:07Z via FH on the same PC.

The records of Fig. 12-(a) show full paths (including a PC name and a user account) as well as contents (including backup *including* and *excluding* folders) of Config.xml files stored in both Win10.E01 and USBStor.E01, indicating that the two different images are associated with each other. In addition, the three records of Fig. 12-(b) describe that USBStor.E01 was connected to Win10.E01 and used as a FH backup storage partition, because there exists the same partition GUID (c2b3c428-7780-4b6c-a785-7a79d4c620ba) in both images. Finally, the (c) record of Fig. 12 is the result of parsing information stored in SOFTWARE hive on the host windows system in Win10.E01. It shows that a storage device (having volume name 'test_8' and drive letter 'E:') shared by a PC named 'WIN-1GIMMFRIB7B' through HomeGroup feature was used for FH backup.

As introduced in the example above, examiners can conveniently analyze FH related configuration information as well as details of backup operations, by utilizing the developed EFIC and the proposed normalization strategy.

[2] https://github.com/chjs207/EFIC-Extracts-FileHistory-IntelligenCe-.

6. Conclusion

This paper proposed a three-step procedure for FH examination: [step-1] identifying FH usage traces, [step-2] determining associated storage devices, and [step-3] extracting backed up file information and analyzing metadata of backed up files. For each step, this work identified various forensic artifacts and described the meaning of each artifact to support FH related forensic activities. In addition, this work analyzed impacts of four potential anti-forensic actions as follows: [action-1] deleting backed up files, [action-2] hiding files through *restore* operation, [action-3] disabling and re-enabling of a backup storage device, and [action-4] disconnecting an active backup storage device.

Based on our findings through various experiments, we developed and open-sourced EFIC, an enhanced FH analysis tool for practical use. This tool provides the ability to extract and normalize FH related useful information, including backed up files, restored files, associated registry values, and disk/partition signatures stored in MBR/GPT.

In Windows, FH is not enabled by default. However, if a user enabled the special feature to back up data, it can be an important subject of DFIR that contains the user's past working history from a digital forensics perspective. In particular, the fact that a network shared drive can be used as a backup storage means that an organization can simply build an effective backup solution with only a built-in feature of Windows. Furthermore, as FH related features are continuously being enhanced according to updates of Windows 10, we expect that the utilization of our findings and developments will gradually increase.

There are several plans for further studies on other features of operating systems that need to be analyzed by integrating a set of data stored on different multiple devices. 'Time Machine' of macOS and 'Your Phone' of Windows are examples of such targets. We hope to contribute to building an enhanced forensic system with a new perspective, in order to respond to the environment where endpoint devices and online services for individual users are ever-increasing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by Korea government (MSIT) (No. 2018-0-01000, Development of Digital Forensic Integration Platform).

Appendix

This appendix contains an extra table for summarizing detailed findings.

Table A1
Summary of File History related artifacts

Source[a]	Type	Target	Format	Description	Sections
Host	File	%LocalAppData%\Microsoft\Windows\FileHistory\Configuration\Config[1,2].xml	xml	FH configurations (e.g., a host machine, a user account, a backup storage, and backup options)	3.2, 3.3, 4.3
	File	%SystemRoot%\System32\Winevt\Logs\Microsoft-Windows-FileHistory-Engine%4WHC.evtx	evtx	Event logs relating to backup starts and ends	3.2, 4.4
	File	%SystemRoot%\System32\Winevt\Logs\Microsoft-Windows-FileHistory-Engine%4BackupLog.evtx	evtx	Event logs relating to 'warning' and 'error' messages	3.2, 4.4
	File	%LocalAppData%\Microsoft\Windows\FileHistory\Configuration\Catalog[1,2].edb	edb	List of backed up files and their metadata	3.4, 4.2
	File	%LocalAppData%\Microsoft\Windows\FileHistory\Configuration\Restore.log	txt	*Restore* operation logs (including size, path, USN, created and modified timestamps)	4.2
	Folder	%LocalAppData%\Microsoft\Windows\FileHistory\Data\	–	Folder for temporarily storing backed up files when a backup storage device is disconnected	4.4
	Reg	HKLM\SYSTEM\?*ControlSet*?\Services\fhsvc\Start	dword	Startup type of *fhsvc* (File History service)	3.2
	Reg	HKEY_USER\?*SID*?\Software\Microsoft\Windows\CurrentVersion\FileHistory\ProtectedUpToTime	qword	Last backup execution time	3.2
	Reg	HKLM\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\RegisteredOwner	string	Registered owner (user) name	3.3
	Reg	HKLM\SYSTEM\?*ControlSet*?\Control\ComputerName\ActiveComputerName\ComputerName	string	Registered computer name	3.3
	Reg	HKLM\SYSTEM\MountedDevices*	binary	Identifiers of mounted storage devices that are disk signatures and partition GUIDs	3.3
	Reg	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\FileHistory\HomeGroup\Target*	various[b]	HomeGroup settings (including the name and URL of HomeGroup-shared backup storage)	3.5
	Reg	HKEY_USER\?*SID*?\Software\Microsoft\Windows\CurrentVersion\FileHistory\TargetChanged	dword	0x00000001 when a configured backup storage device is changed	4.3
Backup	File	\FileHistory\?*UserName*?\?*PCName*?\Configuration\Config[1,2].xml	xml	FH configurations (e.g., a host machine, a user account, a backup storage, and backup options)	3.2, 3.3, 4.3
	File	\FileHistory\?*UserName*?\?*PCName*?\Configuration\Catalog[1,2].edb	edb	List of backed up files and their metadata	3.4, 4.2
	Folder	\FileHistory\?*UserName*?\?*PCName*?\Data\	–	Folder for storing backed up files	3.4
	Folder	\FileHistory\?*UserName*?\?*PCName*?\Data\$OF\	–	Folder for storing backed up files that have paths longer than 146 characters	3.4

[a] Host: a host Windows system's drive, Backup: a backup storage device
[b] This registry key contain string (REG_SZ) and dword (REG_DWORD) type values

References

Dehaviland, O., 2017. Windows 8 file history forensics. https://www.dataforensics.org/windows-8-file-history-forensics/.
Hargreaves, C., Chivers, H., Titheridge, D., 2008. Windows vista and digital investigations. Digit. Invest. 5, 34–48.
Harms, K., 2006. Forensic analysis of system restore points in microsoft windows xp. Digit. Invest. 3, 151–158.
Horsman, G., Caithness, A., Katsavounidis, C., 2019. A forensic exploration of the microsoft windows 10 timeline. J. Forensic Sci. 64, 577–586.
Johnson, W.K., 2013. Journey into windows 8 recovery artifacts. In: Annual ADFSL Conference on Digital Forensics, Security and Law.
Khizra, K., Quba, N., 2014. Windows 8 file history analysis. https://digital-forensics.sans.org/community/summits.
Kobayaashi, M., Suzuki, H., 2018. Reconstruct the world from vanished shadow: recovering deleted vss snapshots. https://www.blackhat.com/us-18/briefings/schedule/.
McKinnon, M., 2017. Autopsy plugin. https://github.com/markmckinnon/Autopsy-Plugins/.
Microsoft, 2018. Homegroup removed from windows 10 (version 1803)). https://support.microsoft.com/en-us/help/4091368/.
plaso, 2019. File history parser of plaso. https://github.com/log2timeline/plaso.

Jisung Choi is a Ph.D Candidate at the School of Cybersecurity, Korea University, Seoul, South Korea. His research interests include digital forensics, incident response, information security, and vulnerability research.

Jungheum Park received a Ph.D. degree from Korea University, Seoul, South Korea in 2014. From Jan. 2015 to Feb. 2019, he was a Guest Researcher for the Computer Forensics Tool Testing (CFTT) project at the National Institute of Standards and Technology's Software and Systems Division, Gaithersburg, MD, USA. Since Mar. 2019, he has been a Research Professor with the School of Cybersecurity, Korea University, Seoul, South Korea. His research interests include security, privacy, and digital forensics.

Sangjin Lee received a Ph.D. degree from Korea University, Seoul, South Korea in 1994. He is a Professor of the School of Cybersecurity, and also a Director of the Digital Forensic Research Center (DFRC), Korea University, Seoul, South Korea. He has served as a president, chair, committee member for many years in a variety of research societies, conferences and workshops. His research interests include security, digital forensics, incident response, steganography, cryptography, and cryptanalysis.