International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 4, June - 2012

# ADS Examiner: Tool for NTFS Alternate Data Streams Forensics Analysis

Sameer H. Mahant [(1)], B.B.Meshram [(2)]

Department of Computer Engineering Veermata Jijabai Technological Institute, Matunga, Mumbai, Maharashtra, India.[(1)(2)]

## Abstract

*The NTFS file system is the most commonly used file system for Microsoft's operating systems. Its Alternate Data Streams (ADS) feature allows the user to hide data in the file system, thus the forensic investigator cannot neglect this fact while doing forensic investigation. The ADS present in deleted file may get overlooked as it is less known in forensic experts. In this paper we have discussed the various methods to hide user's data in ADS, showed the locations where user can create ADS and where should forensic examiner find such hidden evidences. We have also presented how we can modify, delete and retrieve data hidden in ADS and impact of different versions of operating systems on them. Finally we have presented a tool that we have implemented for investigator to find out data hidden in ADS and compared its features with other tools that are available in market.*

## 1. Introduction

Use of data hiding techniques is very common among users and criminals. It is important in computer forensics to find and retrieve the data that is hidden by using various data hiding methods. The most common methods are the ones that are provided by operating system or file system. One such feature of NTFS File System which is used for data hiding is 'Alternate Data Stream'.

This feature had been used by cyber-criminals in their malware to evade detection from antivirus/anti-malware software. The same feature can be used to hide sensitive and illegal data. ADS can store the data without any size limitation (Only limitation is the physical disk size). Alternate Data Streams are not get erased even if the file containing it gets deleted, streams data remains on the disk. This makes forensics analysis of 'Alternate Data Streams' very important as it might contain some evidences in it.

The details of how to create, delete and retrieve ADS, with examples is discussed in the 'Alternate Data Streams' section. This section also highlights the locations where evidences may exist. The "ADS Examiner" section describes the details of tool that we have developed to find Alternate Data Streams on NTFS file system and also discusses the method to find the Alternate Data Stream when the original file to which ADS belongs was deleted.

## 2. Related Work

NTFS is the default file system used by Windows XP, Vista and Windows7 [1]. Aalternate data streams (ADS) is a feature of the NTFS file system that is neither well known nor understood among members of the system administration community. [2] Alternate Data Streams (ADS) are a method of information hiding that is only possible on NTFS file systems. [3] The criminals are using it to hide data because the ADS can hide any size and type of data in NTFS file system. Also the file in which ADS is created does not show the total size of files including ADS, only the size of default stream is displayed in explorer.

There are also legitimate uses of ADS. For example, ADS is used to store summary data and volume change tracking; but each stream should be examined because suspect may also use the common ADS name used by legitimate program to avoid detection. [4]

Windows has native support to create, modify ADS but it does not have ability to view or search the ADS in older operating systems. In Windows Vista and Windows 7 operating systems Microsoft has provided a command line switch (/R) to list the ADS exists in files [2], but it does not support extracting ADS from them.

The author Martini has highlighted that lack of advanced software that is able to detect both simple and compressed Alternate Data Streams has resulted in a less thorough File System examination of Hidden Data. Lack of antivirus programs able to detect a threat contained in Alternate Data Streams raises concerns of malicious users exploiting the ADS could become a leading threat in Computer Security. [5]

The author Brian Carrier focused on presence of ADS in deleted files by saying "Another consideration when recovering deleted NTFS files is to look for additional $DATA attributes" [6]. Method for parsing the binary data stored in disk to analyze the normal files and the deleted files is given by Zhang Kai, Cheng En and Gao Qinquan [7].

## 3. Alternate Data Streams

In this section we will discuss about Alternate Data Streams with examples of their creation, modification, deletion and also how different versions of Microsoft's operating systems have impact on it.

### 3.1 What is ADS?

To support Hierarchical File System (HFS) used by the Macintosh, NTFS file system introduced feature in Windows NT 3.1 called ADS (Alternate Data Streams). File stored on HFS file system used two forks – data fork to keep files data and resource fork to keep metadata of files, such as icons, menus, or dialog boxes. In NTFS file system as we have seen that file is stored as MFT Entry with set of attributes. Each attribute consists of a sequence of bytes called as stream. To support HFS file system, NTFS allowed a file entry to have more than one $DATA attributes in file's MFT Entry. The file's actual data is stored in default (unnamed) $DATA attribute and file's metadata (resource fork of HFS) can be stored in additionally created $DATA attribute. All the additional $DATA attributes that are created for file are known as 'Alternate Data Stream'. Alternate Data Stream is purely a feature of NTFS file system. The windows use ':' to separate filename and ads name. The naming convention is as follows:

Filename.extetsion:AlternateDataStreamName:$DATA

Alternate data streams cannot be viewed using Explorer; they are accessible from the command line (windows vista and later operating systems) and during a disk level analysis [8].

### 3.2Creating ADS

Now we will see how to create ADS using 'command prompt' and 'output redirection operator (>)', which is available in every version of windows. We will also see places where the ADS can exists and how to use it to hide the data.

**3.2.1 Alternate Data Streams in Files.** The simplest way to create ADS in file is using 'echo command' and 'output redirection operator (>)' in windows command prompt. The syntax is as follows:

***Echo text_data >destination_file_path:ads_name***

Example:

***echo Data stored in ADS > F:\myFile.txt:echo.txt***

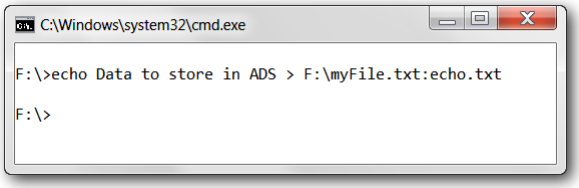


**Figure 1. Creating ADS using echo command**

This command will create an alternate data stream with the name 'echo.txt' in the file 'F:\myFile.txt' and add the text data "Data stored in ADS" into the created ADS.

The NTFS Alternate Data Stream can also store any type of data i.e. image, audio, video, executable files, etc. To store/hide an existing file's data in the ADS, 'type command' is used along with 'output redirection operator (>)'. The syntax is as follows:

***type source_filepath > destination_filepath:ads_name***

Example:

***type F:\myPasswords.txt > F:\myFile.txt:secrets.txt***

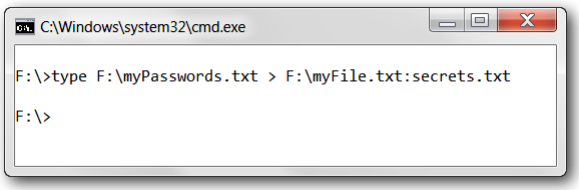


**Figure 2. Creating ADS using type command**

This command will store/hide 'myPasswords.txt' file's data in the alternate data stream of 'myFile.txt' file with the name 'secrets.txt'.

In above example is should be noted that the name of alternate data stream in destination file can be different from source file name. This can be possible because 'type command' does not hide a source file in the alternate data stream of destination files, but it store/copy source file's data as a stream in the destination file's ADS. So even if the name is different from source files, data remains the same. The Alternate Data Stream name may or may not have file extension as it is just a raw content of file.

**3.2.2Alternate Data Streams in Folder.**An Alternate Data Streams can also be created in folders. The folder in NTFS files system does not have any default $DATA attribute, so when ADS is created in folder it is the first $DATA attribute. The method to create ADS in folder is same as the method we have seen earlier for creating ADS in files.

The syntax for creating ADS in folder using 'echo command' and 'output redirection operator' is as follows:

***Echo text_data> destination_folderpath:ads_name***

Example:

***echo "Data to store in ADS" > F:\myFolder:echo.txt***

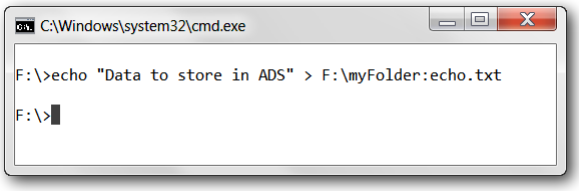


**Figure 3.Creating ADS in Folders(echo command)**

This command will create an alternate data stream with the name 'echo.txt' in the folder 'C:\myFolder' and add the text data "Data to store in ADS" (including " ") into the created ADS.

The syntax for creating ADS in folder using 'type command' and 'output redirection operator' is as follows:

***type source_filepath > destination_folder:ads_name***

Example:

***type F:\myPasswords.txt > F:\myFolder:secrets.txt***

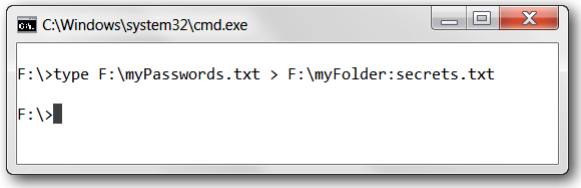


**Figure 4. Creating ADS in Folders (type command)**

This command will store/hide 'myPasswords.txt' file's data in the alternate data stream of 'myFolder' folder with the name 'secrets.txt'.

**3.2.3Alternate Data Streams in Partition.**In NTFS file system the ADS can also be present at partition level. Not all forensic examiners know this fact about ADS. In files and folder, when ADS is created new $DATA attribute is added to their MFT Entry. Then how ADS can be present at partition level? Where ADS of partition gets stored?

To answer above questions we have to understand that in NTFS files system every object is considered as file. One of the NTFS metadata file is named '.' (Dot), and it is 5th entry in MFT. This metadata file is considered as the root for all other files and folder. So when ADS is created at partition level, new $DATA attribute is added in the '.' (Dot) metadata file's MFT Entry.

The syntax for creating ADS in partition using 'echo command' and 'output redirection operator (>)' is as follows:

***echo text_data>partition_name::ads_name***

Example:

***echo ADS in Partition > F::partADS.txt***

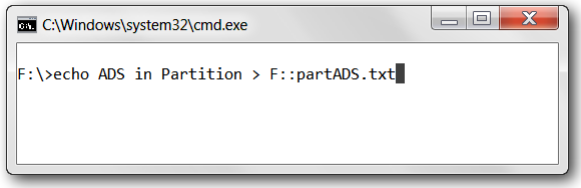


**Figure 5. Creating ADS in Partition (echo command)**

This command will create an alternate data stream with the name 'partADS.txt' in the 'F:' partition and add the text data "ADS in Partition" into the created ADS.

The syntax for creating ADS in partition using 'type command' and 'output redirection operator' is as follows:

***type source_file_path >partition_name::ads_name***

Example:

***type F:\myPasswords.txt >F::secrets.txt***

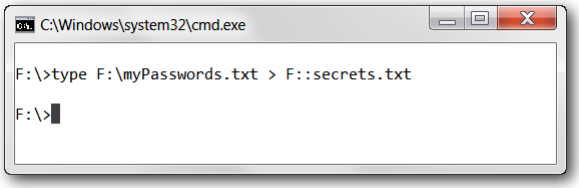


**Figure 6. Creating ADS in Partition (type command)**

This command will store/hide 'myPasswords.txt' file's data in the alternate data stream of 'C:' partition with the name 'secrets.txt'.Note that while creating ADS at partition level we have used double colons [::] before the name of ADS.

Microsoft's windows XP, Vista and windows 7 support creating ADS in files, folder and partition.

## 3.3 Deleting ADS

Alternate Data Streams (ADS) are a method of information hiding that is only possible on NTFS file systems [3], therefore simplest method to delete ADS from file or folder is by copying it to another not-NTFS file system. Since ADS is a feature of NTFS it can recognize ADS but other file systems do not. When we copy a files or folder to non-NFTS files system, only the main $DATA streams is get copied and other streams or lost.

To destroy the sensitive data stored in ADS then we can 'echo' <any character> or 'type' any file with garbage data in it with the same ADS name to the destination file which contain ADS. It will overwrite the contents which was previously stored in Alternate Data Stream.Example:

***type C:\blank.txt > C:\myFile.txt:secrets.txt***

Now previously stored contents from passwords.txt get replaced by contents in blank.txt, which may be garbage data or no data at all. This will erase the data stored in Alternate Data Stream but not the existence of Alternate Data Stream i.e. extra $DATA attribute still be present in MFT Entry with garbage data.

To completely remove existence of ADS i.e. to remove extra $DATA attribute, windows does not have any option, we have to use tools such as ADS Spy, Streams.exe or Hijackthis to delete the Alternate Data Streams. Another way is by manualy modifying data by getting low level access to disk's containts.

## 3.3 Retrieving Data from ADS

We have seen how to store contents in Alternate Data Streams, now we will look at how to get back the contents from the Alternate Data Stream. We will discuss how to directly run/execute the data stored in Alternate Data Streams without copying to other location.

We can retrieve the hidden text file contents from the Alternate Data Stream using the 'more' command, input redirection operator (<) and output redirection operator (>) present in windows operating system. The syntax is as follows:

***more< "full_path_to_ads" > destination_file_name***

Example:

***more< myFile.txt:secret.txt > recoveredSecret.txt***

In the above example the 'more' command takes the input as contents of stream 'myFile.txt:secret.txt' using '<' operator and then finally redirect it to 'recoveredSecret.txt' using '>' operator. We cannot use 'type' command because the 'type' command doesn't accept stream syntax.

This method works only for text data. In windows there is no support/command exists to retrieve non-text file's content. We have to use other tools like AlternateStreamView.Although we can not retrieve non-text content store Alternate Data Stream. We can execute them with appropriate applications. Applications like Adobe Reader, Microsoft's paint, notepad, etc. can work with data stored in Alternate Data Streams. For example:

- If an image file's contents are stored in ADS we can open it directly in 'Paint' application available in Microsoft's Windows OS. The syntax is as follows:

  ***mspaint "full_path_to_ads"***

- The executable file's contents stored in ADS can be executed using 'start' command available in Windows. The syntax is as follows:

  ***start "full_path_to_ads".***

  If you are in same directory in which file with ADS is present that you want to execute without using full path to file then you have to prepend the '.\' before the filename:adsname. Example:

  ***start .\filename:adsname***

The execution of ADS was possible in Windows XP, but in Windows Vista and Windows 7 it is no more possible to execute files from ADS using 'start' command. Attempting to do so give following error message:

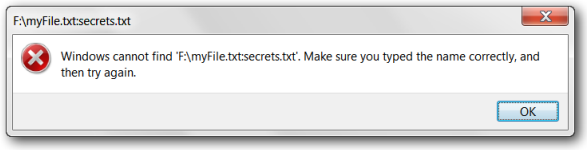


**Figure 7. Error message when opening file hidden in ADS on Windows 7**

### 3.5Dark Side of ADS

Alternate Data Stream feature of NTFS file system was originally designed to increase compatibility with Macintosh systems, but bad guys have used ADS to hide tools, even as part of rootkits, because a file held in the ADS of another file has no icon of its own and is not displayed to the user by Windows explorer; however, a user can still access a file placed in ADS and even run it directly from the ADS without having to extract it from its hidden location [9]. Alternate data streams facilitate the attachment of data to any file or directory on an NTFS file-system, such that the data is not actually part of the file content and nor is its existence apparent through normal file and directory API calls (e.g., this can be used as a primitive mechanism to hide files or data). For example, under Win-dows XP, information entered in the Summary tab of a file's Properties dialog box are saved as alternate data streams attached to the file. Creating, reading and writing alternate data streams is very easy for a program or user to do, and not so easy to detect without using special tools that specifically search for alternate data streams [10].

Not all antivirus utilities scan ADS or do not do so by default. Therefore, malware that is dropped onto a system in ADS might not be detected or removed/quarantined by the antivirus application. For example: In 2000, Benny and Ratter released a virus named W2K.Stream that used ADS. The virus would infect a file, replace it, and then copy the original file into ADS.

After all of above discussion it is clear that Alternate Data Streams in NTFS files system can not be neglected while performing forensic analysis. This motivated us to make a tool (described in next section) that can search ADS in all possible locations and help investigator to perform forensics analysis on ADS.

## 4. ADS Examiner Tool

In this section we will discuss the implementation details of the tool. The block diagram of tool is given in **Figure 8.**

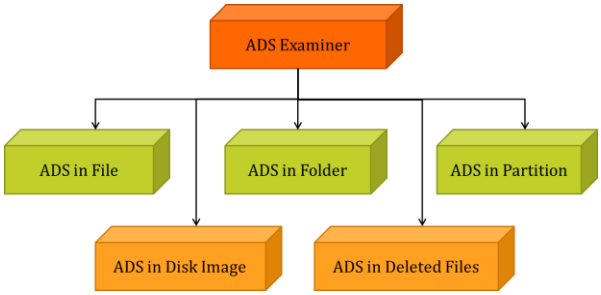


**Figure 8. Block diagram of ADS Examiner tool**

The tool has five different functions to search the Alternate Data Streams as per the suspected location of ADS. Tools can search ADS in multiple files, ADS in folder, ADS exists at partition level. These functions will be discussed in the paper. Advance functions to find ADS in deleted files of partition and ADS in given disk image (.dd format) are also available in tool to help forensic investigator.

### 4.1Finding ADS in Files (On Live System):

To find ADS in files we used FindFirstStreamW() and FindNextStreamW() file management functions. These functions are available in "Kernel32.dll" file. According to Microsoft[11]:

FindFirstStreamW() function enumerates the first stream with a ::$DATA stream type in the specified file or directory. The syntax and description of parameters is given below:

```
HANDLE WINAPI FindFirstStreamW(
__in        LPCWSTR lpFileName,
__in        STREAM_INFO_LEVELS InfoLevel,
__out       LPVOID lpFindStreamData,
__reserved  DWORD dwFlags
);
```

The first parameter (lpFileName) is a fully qualified name of file i.e. file's complete path. The second parameter (InfoLevel) is the information level of the returned data. There is only one level available for now named 'FindStreamInfoStandard', having numerical value of 0 (Zero) which is used to return data in a WIN32_FIND_STREAM_DATA structure. The third parameter (lpFindStreamData) is a pointer to a buffer that receives the file stream data. The fourth parameter (dwFlags) is reserved for future use and must be zero.

FindNextStreamW () function continues a stream search started by a previous call to the FindFirstStreamW() function. The syntax and description of parameters is given below:

```
BOOL WINAPI FindNextStreamW(
  __in   HANDLE hFindStream,
  __out  LPVOID lpFindStreamData
);
```

The first parameter (hFindStream) is search handle returned by a previous call to the FindFirstStreamW function. The second parameter (lpFindStreamData) is a pointer to the WIN32_FIND_STREAM_DATA structure that receives information about the stream.

WIN32_FIND_STREAM_DATA contains the information of stream found by FindFirstStreamW() or FindNextStreamW() function. It stores the stream size in bytes and name of the alternate streams in "**:streamname:$streamtype**" format.

The structure definition is given below:

```
typedef struct _WIN32_FIND_STREAM_DATA {
  LARGE_INTEGER StreamSize;
  WCHAR         cStreamName[MAX_PATH + 36];
}
```

We have to use both these function to find the ADS present in files and folders. The psudocode for the SearchAlternateDataStreams_InFile() function is given below:

**Function Name:** SearchAlternateDataStreams_InFile(inputFilePath)

**Input:** inputFilePath (file's full path)

**Variables:**Output_ADS_List; is a list of type WIN32_FIND_STREAM_DATA

CurrentStream; is a variable of type WIN32_FIND_STREAM_DATA

**Processing**:

```
// Get handle returned by FindFirstStreamW() function into handle variable
Handle = FindFirstStreamW(inputFilePath, 0, CurrentStream, 0);

If Handle is invalid
Display the error message and return.
Else
Do
Add CurrentStream to Output_ADS_List;
While ( FindNextStreamW(Handle, CurrentStream) returns valid Handle)
```

**Output**: Output_ADS_List which will contain information about file's all streams

The SearchAlternateDataStreams_InFile() function takes input as 'inputFilePath' which is the full path of file in which you want to find ADS. The function first call the FindFirstStreamW() function with parameters 'inputFilePath' and 'CurrentStream' structure of type WIN32_FIND_STREAM_DATA to get the 'Handle' to the file for further processing. If the returned 'Handle' is invalid it will display error message and return. Otherwise it will add the CurrentStream which contains information of current file stream, into the Outout_ADS_List. Then the FindNextStreamW() function is called with the current file handle 'Handle' to enumerate other streams that may exists in files. In the output we will get Output_ADS_List of type WIN32_FIND_STREAM_DATA, which contains information about each stream, exists in file.

### 4.2Finding ADS in Folder (On Live System):

Finding ADS in folder is not much different than we saw in files. Difference is that the folders does not have any default data stream like files, therefore the above function will always throw exception in case of normal folders which does not contain ADS, because the FindFirstStreamW() will not find any stream and will return invalid handle. To solve this problem we have to continue the processing of folder even if FindFirstStreamW() does not return valid file handle. The psudocode for the implemented function SearchAlternateDataStreams_InFolder() is as below:

**Function Name:** SearchAlternateDataStreams_InFolder(inputFolderPath)

**Input:**inputFolderPath (folder's full path)

**Variables:** Output_ADS_List; is a list of type WIN32_FIND_STREAM_DATA
CurrentStream; is a variable of type WIN32_FIND_STREAM_DATA

**Processing**:

```
// Get handle returned by FindFirstStreamW() function into handle variable
Handle = FindFirstStreamW(inputFilePath, 0, CurrentStream, 0);

If Handle is invalid
Continue processing   // because Folder does not have default $DATA stream
Else
   Do
Add CurrentStream to Output_ADS_List
   While ( FindNextStreamW(Handle, CurrentStream) returns valid Handle)
```

**Output**: Output_ADS_List which will contain information about file's all streams

### 4.3Finding ADS at Partition Level (On Live System):

To find ADS at Partition Level we do not have to write separate code because the previous code of 'Finding ADS in Folder' also finds ADS at Partition Level when we give input as Partition Name. This is possible because the when we give Partition Name as input it actually points to root directory of drive.

### 4.4Finding ADS in Deleted Files and in Disk Image:

Finding ADS in Deleted files or in Disk Image is very different from what we have seen in above section by using the API provided by Microsoft. To find the ADS in Deleted Files requires low level access to disk and the understanding of NTFS file system structural and operational details. The general flowchart for finding ADS in Deleted Files is given in **Figure 10.**

Description of flowchart is given below:

1. Get the Master Files Table's first MFT Entry's sector address. To get this address we have to first process the MFT's $DATA attribute to get its clusters and then convert them to sector addresses for processing.
2. Set the file pointer position to the start of sector address and read the 1024 bytes (size of MFT entry) into a buffer.
3. Check the MFT entry's signature i.e. first four bytes of buffer. If it is equal to "FILE" continue processing else go to next MFT entry.
4. Once we verified Entry's signature we have to fixup the buffer before further processing. Microsoft uses fixup values at the end of each sector (512 bytes) if the object stored (in this case MFT Entry) requires more than one sector. This value is used for checking integrity of MFT Entry.
5. After that process the MFT Entry and store all files attributes in Attribute_List.
6. Check if the MFT Entry is for file and the file is deleted. (This is done using the value located at offset 22-23 in buffer.) Then get the count of $DATA attributes and add its info to Output_List if count is greater than 1.
7. Check if the MFT Entry is for folder and the folder is deleted. Then get the count of $DATA attributes and add its info to Output_List if count is greater than 0.
8. If the current entry is last MFT Entry then stop, otherwise get the next sector address and continue search procedure.

To help the examiner in finding ADS for forensics investigation the tool provides automated capability to find ADS in the files which are deleted by suspect with $STANDARD_INFORMATION and $FILENAME attribute's time information. It also supports '.dd' disk image format of images created using 'dd command'.

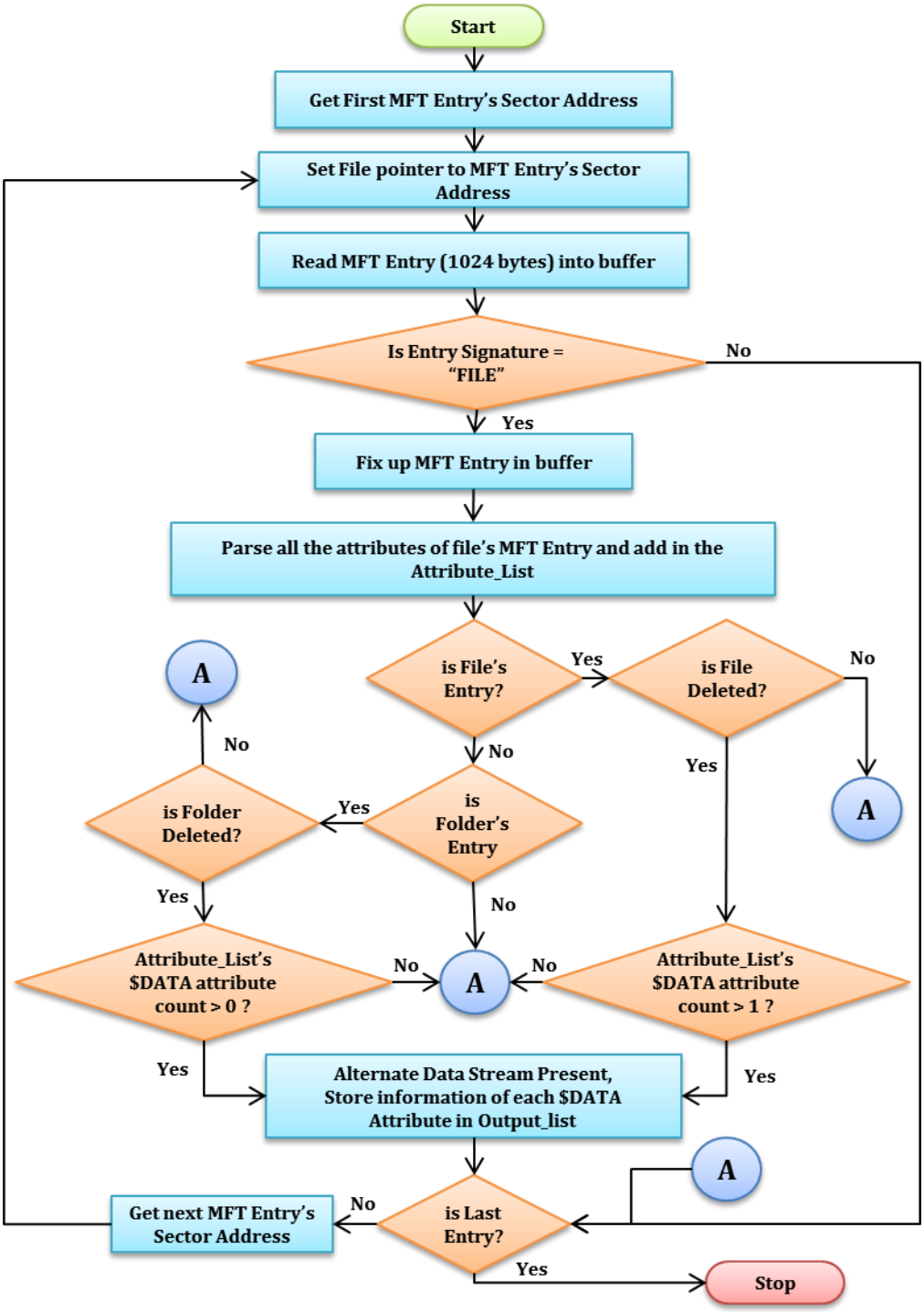


**Figure 9. Flowchart for finding ADS in deleted files**

### 4.4 Alternate Data Stream Tools Comparison

In this section various tools available to find ADS in NTFS files system are compared with our own developed 'ADS Examiner' tool. The comparison is done based on tools capability find ADS in various locations, support for copying ADS for further examination, support for finding ADS in deleted file and disk image, Graphical User Interface availability for tool is also considered.

The result of comparison is show in **Table1.**

**Table 1. Comparison of ADS Examiner with other ADS finding tools**

| Tool Name | Finding Alternate Data Stream in(on Live System) | | | Copying ADS | Finding ADS in Deleted Files | Finding ADS in Disk Image | Graphical User InterfaceAvailable |
|---|---|---|---|---|---|---|---|
| | File | Folder | Partition | | | | |
| ADS Examiner v1.0 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ADS Spy v1.11 | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ |
| AlternateStreamView v1.32 | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ |
| LADS v4.0 | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| Streams v1.56 | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| Marx NFTS ADS Viewer v2.0 | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Sfind v2.0 | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |

Screenshot of ADS Examiner tool with sample results is given in **Figure 10**which is shownbelow:

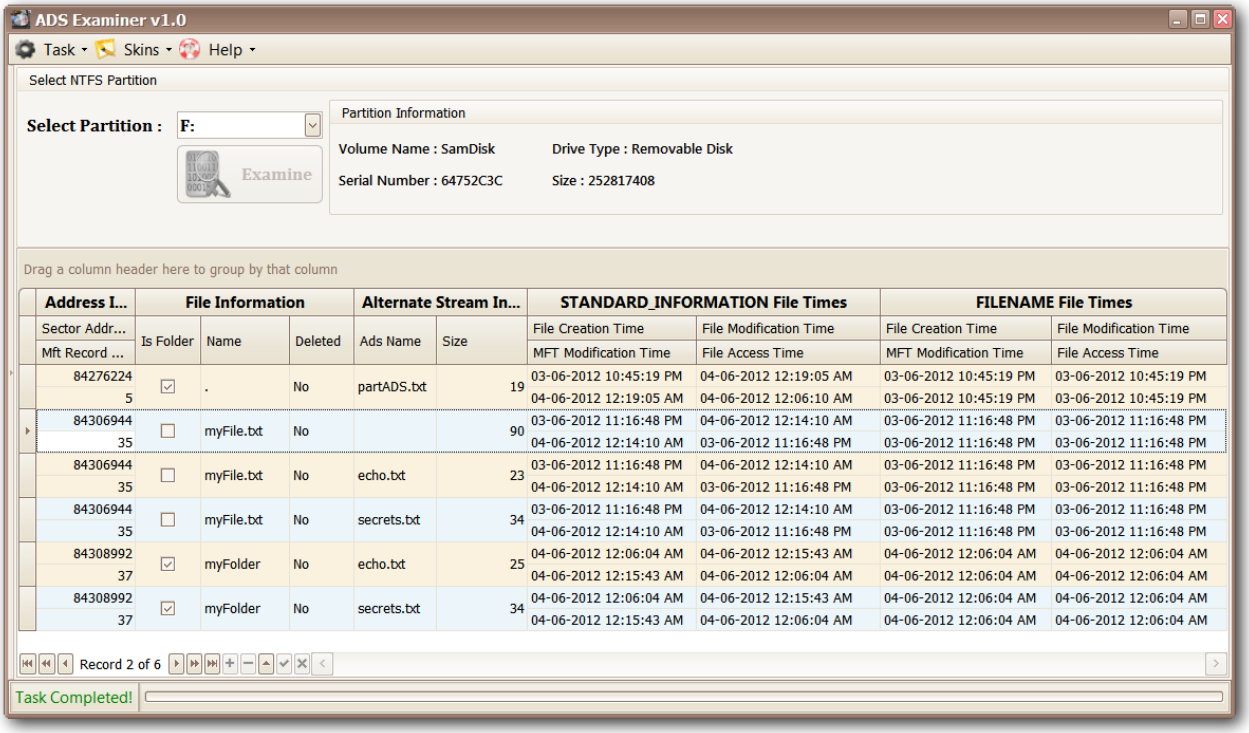


**Figure 10. ADS Examiner tool showing streams information with time**

## 5. Conclusion

The Alternate Data Stream feature of NTFS file system has been used by criminals from long time. Also the native commands available in windows to create and modify Alternate Data Streams encourage criminals to use them. Though the newer Operating systems like Vista and Windows 7 has blocked the execution of data in Alternate Data Stream using 'start' command, the other applications can still access and use/execute data stored in Alternate Data Streams.

The Windows Vista and Windows 7 only has feature to view data stored in Alternate Data Stream, but it cannot extract the data. Forensic examiners required to use tools which can have this facility. To satisfy this need we have developed the "ADS Examiner Tool" which can find Alternate Data Streams at various locations including files, folders, and partitions. It also extracts the data from Alternate Data Streams. The disk image support is provided for dead forensic analysis.

The comparison of ADS Examiner with other tools also showed that the tool is useful for the forensic investigator in deep forensics investigation of Alternate Data Strems present at various locations in the NFTS file system.

## 6. References

[1] D. D. Hayes, V. Reddy and S. Qureshi, "The Impact of Microsoft's Windows 7 on Computer forensics examinations," in *Applications and Technology Conference (LISAT), 2010 Long Island Systems*, 2010.

[2] H. Carvey, "Windows Forensic Analysis DVD Toolkit 2E," in *Windows Forensic Analysis DVD Toolkit 2E*, Burlington, MA, Syngress Publishing, Inc., 2009, pp. 299-320.

[3] J. Davis, J. MacLean and D. Dampier, "Methods of Information Hiding and Detection in File Systems," in *SADFE 2010, Fifth IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering*, Oakland, CA, USA, 2010.

[4] C. K. Wee, "http://www.forensicfocus.com/hidden-data-analysis-ntfs," 2006. [Online]. Available: www.forensicfocus.com/downloads/ntfs-hidden-data-analysis.pdf. [Accessed 2 6 2012].

[5] A. I. Martini, A. Zaharis and C. Ilioudis, "Detecting and Manipulating Compressed Alternate Data Streams in a Forensics Investigation," in *WDFIA, Third International Annual Workshop on Digital Forensics and Incident Analysis*, Malaga, Spain, 2008.

[6] B. Carrier, File System Forensic Analysis, Addison Wesley Professional, 2005.

[7] Z. Kai, C. En and G. Qinquan, "Analysis and Implementation of NTFS File System Based on Computer Forensics," in *The Second International Workshop on Education Technology and Computer Science*, Wuhan, Hubei, China, 2010.

[8] B. Sheldon, "Forensic Analysis of Windows Systems," in *Handbook of Computer Crime Investigation Forensic Tools and Technology*, E. Casey, Ed., Great Britain, Academic Press, 2003, pp. 133-166.

[9] R. D. Pittman and D. Shaver, "Windows Forensic Analysis," in *Handbook of Digital Forensics and Investigation*, E. Casey, Ed., United States of America, Elsevier, 2010, p. 221.

[10] F. Mirza, "Looking for Digital Evidence in Windows," in *Biometrics and Security Technologies, 2008. ISBAST 2008. International Symposium*, Islamabad, 2008.

[11] "http://msdn.microsoft.com/en-us/library/windows/desktop/aa364424(v=vs.85).aspx," Microsoft, 6 4 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windows/desktop/aa364424(v=vs.85).aspx. [Accessed 2 6 2012].