Design and Development of Improved Stealth Alternate Data Streams

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Ruhi Mahajan
(Roll No. 801232019)

Under the supervision of:

Dr. Maninder Singh
Associate Professor
CSED

Mr. Sumit Miglani
Assistant Professor
CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

July 2014

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, *"Design and Development of Improved Stealth Alternate Data Streams"*, in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Maninder Singh* and *Mr. Sumit Miglani* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Ruhi Mahajan
Ruhi Mahajan

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Maninder Singh)
Associate Professor, CSED
Thapar University, Patiala

(Mr. Sumit Miglani)
Assistant Professor, CSED
Thapar University, Patiala

Countersigned by

(Dr. Deepak Garg)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

ACKNOWLEGEMENT

The successful completion of any task would be incomplete without accomplishing the people who made it possible and whose constant guidance and encouragement secured the success.

First of all I wish to acknowledge the benevolence of omnipotent God who gave me strength and courage to overcome all obstacles and showed me the silver lining in the dark clouds.

With the profound sense of gratitude and heartiest regard, I express my sincere feelings of indebtedness to my guides Dr. Maninder Singh, Associate Professor, Computer Science and Engineering Department and Mr. Sumit Miglani, Assistant Professor, Computer Science and Engineering Department, Thapar University for their positive and excellent guidance, constant encouragement, keen interest, invaluable co-operation, generous attitude and above all their blessings have been persistent source of inspiration for me.

I am grateful to Dr. Deepak Garg, Head of Department, and Dr. Ashutosh Mishra, P.G. Coordinator, Computer Science and Engineering Department, Thapar University for the motivation and inspiration that triggered me for this thesis.

Last but not the least I would like to express my heartfelt thanks to my parents and my friends who with their thought provoking views, veracity and whole hearted co-operation helped me in doing this thesis.

Ruhi Mahajan

M.E (CSE)

801232019

ABSTRACT

With increase in usage of Internet, there is a greater need of protecting sensitive information. Various data hiding techniques are available for protecting data from unauthorized users. Alternate Data Streams is one of the possible ways for data hiding in New Technology File System of Windows. It was introduced to make Windows NTFS compatible with HFS file system of Macintosh. Alternate data streams is an important feature of New Technology File System but some consider it as a vulnerability which can be exploited for hiding malicious files like rootkits, virus, backdoors etc and for getting access of victim's system. So Alternate data streams is both a feature and vulnerability of NTFS.

This thesis explains what exactly alternate data streams are, what are its requirements and functionalities. A demonstration explaining how hackers can exploit Alternate data streams or ADS for getting access of a system is shown. Its main focus is on explaining Stealth ADS that provides the important functionalities like creation, detection, and deletion of ADS. All possible ways of hiding sensitive information and techniques for detecting and removing ADS are also explained. An approach for scanning the detected ADS in the system for the presence of malicious files is also explained.

With time and advancement in antivirus technologies, ADS are now detected by various antivirus and thus with a aim to enhance its stealth and to bring it back into action, efforts has been made to bundle ADS technology with an external encoder which eventually improves its stealth to great extent. To prove this point, a comparative analysis indicating increase in stealth of alternate data stream by adding encoder has been performed. This comparison is done between existing metasploit encoders and Stealth ADS encoder. Also comparison of Stealth ADS with the existing software is done for proving the efficiency of Stealth Alternate Data Streams.

TABLE OF CONTENTS

CERTIFICATE i

ACKNOWLEDGEMENT ii

ABSTRACT iii

TABLE OF CONTENTS iv

LIST OF FIGURES vi

LIST OF TABLES vii

CHAPTER 1: INTRODUCTION 1

1.1 Network Security 1

1.1.1 Common Terminologies in Network security 2

1.1.2 Security Threats 3

1.1.3 Network Security Tools 4

1.2 Data Hiding 4

1.3 Thesis Online 5

CHAPTER 2: LITERATURE REVIEW 6

2.1 File System 6

2.1.1 File System in Windows 6

2.2 Architecture of NTFS 6

2.3 NTFS Specific Data Hiding Method 9

2.4 Alternate Data Streams 13

2.4.1 Features of Alternate Data Streams 13

2.4.2 Pros and Cons of ADS 14

2.5 NTFS Streams 15

2.6 Hierarchical File System 16

2.7 Requirement of ADS 17

2.8 Detecting ADS in Forensic Investigation 18

2.8.1 Forensic Challenges 19

2.9 Legitimate use of ADS 22

2.10 Windows app that do not recognize ADS 25

2.11 ADS Vulnerability 26

CHAPTER 3: PROBLEM STATEMENT **29**
3.1 Gaps in study 29
3.2 Problem Statement 29
3.3 Objectives 30

CHAPTER 4: IMPLEMENTATION **31**
4.1 Stealth ADS 31
4.1.1 Creation of ADS 31
4.1.2 Detection of ADS 33
4.1.3 Deletion of ADS 35
4.1.4 Encoder 36
4.1.5 Virus Scanner 38
4.2 Python 39
4.2.1 Features of Python 39
4.2.2 Python Modules used 40

CHAPTER 5: EXPERIMENTAL RESULTS **43**
5.1 Improving Stealth of ADS using Encoder 43
5.2 Comparison of SADS encoder with Existing Encoders 46
5.3 Exploitation of ADS 47
5.4 Comparison of SADS with existing Software 50

CHAPTER 6: CONCLUSION AND FUTURE SCOPE **52**
6.1 Conclusion 52
6.2 Future Scope 52

REFERENCES **54**
LIST OF PUBLICATIONS **57**

LIST OF FIGURES

Fig 2.1: NTFS architecture 7

Fig 2.2: MFT record for a small files 7

Fig 2.3: Analysis of data hidden in added clusters 10

Fig 2.4: Analysis of hidden data in files. 11

Fig 2.5: Visibility of ADS in NTFS and non-NTFS environments 12

Fig 2.6: Analysis of hidden data in directory 12

Fig 2.7: Comparing architecture of FAT and NTFS 16

Fig 2.8: Architecture of file in HFS 17

Fig 2.9: Compatibility of Windows and HFS 18

Fig 2.10: Showing adding ADS while compressing a file 22

Fig 2.11: Contents of file containing Zone.Identifier 23

Fig 2.12: Dialog box asking for user permission for executing a file 24

Fig 2.13: Showing capability of deleting Zone.Identifier attached to a downloaded file 24

Fig 4.1: Flow chart showing the control flow of creation module 32

Fig 4.2: Stealth ADS showing hiding a text file behind an image 32

Fig 4.3: Opening hidden file text.txt 33

Fig 4.4: Flowchart showing control flow of detection module 34

Fig 4.5: SADS showing all ADS present in directory E:\ads 34

Fig 4.6: Flowchart showing control flow of deletion module 35

Fig 4.7: SADS showing deletion of selected ADS 36

Fig 4.8: Encoding and decoding of ADS file 37

Fig 4.9: Flowchart showing control flow of virus scanner module 38

Fig 5.1: Graphical representation of results obtained from virustotal 45

Fig 5.2: Comparing Encoders on the basis of detection rate and encoding time 46

Fig 5.3: Creation of a malicious file using metasploit framework 48

Fig 5.4: Downloading backdoor_win.exe on victim's machine 48

Fig 5.5: Hiding Malicious file in ADS namedbackdoor.exe 48

Fig 5.6: Attacker listening to port so as to get shell 49

Fig 5.7: Showing attacker has got access of victim's system 49

Fig 5.8: Getting shell by executing malicious ADS file 50

LIST OF TABLES

Table 2.1: Description of various attributes in NTFS 8
Table 4.1: Description of various Python Modules 40
Table 5.1: Comparative results by scanning files using Virustotal site 43
Table 5.2: Comparing detection rate of different encoders for three malicious files 47
Table 5.3: Comparison of Alternate data stream tool against already available Software 50

CHAPTER 1
INTRODUCTION

With the increasing role of Information Technology in every field of life whether it is business, education, banking, etc the connection with Internet is increasing day by day. People share a lot of information over Internet. So with the increasing information sharing, there is greater need of securing the information. There are various vulnerabilities present in the applications that we are using today. So they provide an easy way for people having evil intentions to fulfil their desires. Thus it has become difficult for network administrators and network security personnel to secure data from various types of attacks. So security issues should be taken into account in every area. Data or information needs to be secured from unauthorized access; this can be done by hiding the data. Data hiding makes data unavailable for unintended users. There exist different ways and software in the market for hiding data but for them we need to either purchase them or to download them thus making it somewhat cumbersome task. There exist a data hiding technique in Windows having NTFS file system. This technique is known as Alternate Data Streams. It is a feature of NTFS file system that can be effectively used for hiding purpose but some attackers consider it as vulnerability and exploit it for their own benefit.

1.1 Network Security

It is a branch of computer science that deals with securing computer network. It is the responsibility of the network administrator to implement all the security policies that are needed for protecting a network. Network administrator should prevent any unauthorized access to data over network. Network security implies that if two systems are communicating over Internet then no third user should interfere in their communication. So network security is mainly based on securing the information over network. The main goals of Information Security are [1, 2]:

1) Confidentiality: It means saving or protecting information from illegal access that is protecting information from unauthorised access. In case of banks, the confidential information is account number details or credit card details of customers which should not be leaked. If such information does not remain confidential in bank then it will lose its customers. So confidentiality is very important for securing information. To maintain confidentiality of data, encryption is used. With encryption, only the

authorised user will be able to access the data. So data will remain confidential. It also involves enforcing permissions for accessing files containing sensitive information.

2) Integrity: It implies maintaining the accuracy of data. Data should not be changed or altered by any unauthorised person. In case of banks, if account number of a person is changed then it will create a big problem. It means if data is transferred from source then it should not be changed before reaching the destination. Data integrity is maintained by hashing the data and comparing it with the hash of the actual message. This comparison is done at the receiver site.

3) Authenticity: It means identifying the originality of the data. It ensures that the two parties involved in communication are genuine. Digital Signatures are used now-a-days for insuring authenticity. Authenticity means that only authorised user should get access to the information.

4) Availability: It means that the information should be available to authorised user whenever it's needed. Suppose an organisation data is lost due to system failure then there should be backups available so that they can recover their data.

1.1.1 Common Terminologies used in Network Security

The various commonly used terminologies are [3, 4] explained below:

1) Threat: Threat is a potential danger that can lead to a system compromise. Threat is a fear that attacker will get into the system using the known vulnerabilities in the system. Threat can be deliberate as well as accidental also.

2) Exploit: Exploit is a kind of code that is directed used by the attacker for hacking purpose by breaking the system security and exploiting the known vulnerability. Exploit can be different for different vulnerabilities and it is not necessary that an exploit working for one machine (Operating System) will work for other machine (different Operating System) even if they share the common vulnerability.

3) Vulnerability: In layman`s language vulnerability means any loophole which can be misused by hacker to get into the system. These Loopholes are the entry points for the hackers. Consider a lock which can be open by some other key. So the fact that the lock can be opened by any key is a vulnerability or weakness. Vulnerability occurs due to the negligence either from administrator side or from coder's side.

1.1.2 Security Threats

The major security threats of information security are explained below [2]:

1) Hacking: It can be defined as an unlawful act of getting access of somebody's computer and owner of the system has no knowledge regarding this. The person doing such activity is known as hacker. Once a person gets access to a system then that person can steal all sensitive information and can also alter data present in the system. After doing changes hackers remove traces so that owner of the system cannot know about the intrusion.

2) Viruses or Worms: These are the programs that affect the functionality of a system. Although both perform the same task, but there is a little difference between two-virus needs a carrier for its transfer to victim machine whereas worms do not. Both replicate themselves at the victim's system [5].

3) Trojan Horse: These programs were initially used by system administrators for administration that is for controlling their remote workstations. But later on they were used by hackers for illegal purpose like getting access of victim's machine, stealing sensitive information from victim's machine. These can be easily installed on target machine by sending an attachment in email.

4) Spoofing: Meaning of spoofing is imitating. Thus it means fooling authorised user that the information source is a legitimate system. But actually it is not happening [6]. Some of its common prevention techniques are engress filtering, egress filtering and attempts to stop the packets at the destination, traceback [7].

5) Sniffing: It means intercepting data. It is software or a program that reads all the data passing over a network without altering it. It is like communication is taking place between two persons and third person is listening without their knowledge. The protocols that are vulnerable to this are those that send data and passwords in clear text. Such protocols are: SMTP, HTTP, FTP etc. The only solution to prevent this is sending data in encrypted form.

6) Denial of Service: This attack does not imply stealing data but it makes target system unavailable for usage. The target system is not able to provide service to legitimate users. Ping command can be used for performing this attack. By flooding large number of Ping messages to a server, the server is not able to handle so much

load and crashes. Normally attackers use large number of zombies for performing this attack and that is known as distributed denial of service (DDOS) attack [8].

1.1.3 Network Security tools

The various security tools available in market for protecting system against security threats are [9]:

1) Antivirus: The antivirus software detects potential security threats by matching the signatures of the files with the signatures stored in their database. So the antivirus needs to be properly updated so that they can perform perfectly.

2) Secure Network Infrastructure: In the network, switches and routers should be used as they provide functionality of both hardware and software and provides services like intrusion protection, security management etc.

3) Virtual Private Network: In this network when data is transferred between two nodes, it is transferred in encrypted form. Thus data integrity is maintained. So, chances of intrusion decrease in such networks.

4) Identity Services: These services help in determining the identity of the user that is whether user is authorised or not or identifying the source of data. This includes services like digital signatures, passwords etc.

5) Vulnerability Assessment: Various tools are available in market that shows the presence of various vulnerabilities in user's system. Once the vulnerabilities are known, user should immediately patch them before they get exploited by an attacker. Some of popular Vulnerability Assessment tools are Acunetix, N-Stalker, IBM Rational AppScan etc and each has their merits and demerits based on their accuracy and detection rate [10].

1.2 Data Hiding

Data hiding or information hiding basically deals with protecting sensitive data from unauthorised access [11]. It makes data invisible to unintended users. Information hiding is also used by hackers or criminals for hiding their criminal activities. Thus Information hiding is done by both criminals and normal users. There are various techniques available for data or information hiding. These are [12]:

1) Non- Physical data hiding: It is related to hiding in digital form. The various forms of non-physical data hiding are Steganography, Cryptography and Watermarking. Steganography implies that sensitive data should be hidden in a normal message [13]. Cryptography implies protecting data from unauthorised users by encrypting data in unreadable form known as cipher before sending. The difference between Steganography and cryptography [14, 15] is that in Steganography, data is sent in plain text (embedded in other data) not in encrypted form like in cryptography. Watermarking [14] technique is basically used for proving copyright information of a document. This is done by adding pattern bits in audio, video files. The bit patterns used for defining copyright information should be made invisible by scattering them so that they cannot be manipulated.

2) File System data hiding: It is a form of physical data hiding that involves hiding data in storage locations like hard disk, file systems. Various methods are available for hiding data in file system. These are hidden files or folders, slack space, deleted files, deleted or hidden partitions, bad clusters, Alternate data streams etc.

1.3 Thesis Outline

Thesis has been organised into six chapters:

Chapter2 is Literature Survey that includes explanation of new technology file system (NTFS), data hiding techniques in NTFS, HFS, reasons behind introducing alternate data streams in NTFS and detailed explanation of ADS.

Chapter3 is Problem Statement that includes explaining various gaps in existing system and discussing problem statement.

Chapter4 is Implementation that includes detailed explanation of stealth Alternate Data Stream including creation, deletion, detection, encoding and scanning.

Chapter5 is Results in which comparison of existing encoder and created encoder is performed, also comparison of stealth Alternate data stream is performed with available software and demonstration of attack that can be performed using Alternate data streams is given.

Chapter6 is about conclusion and future Scope which includes summarising thesis and provides future research direction.

CHAPTER 2

LITERATURE REVIEW

2.1 File System

A file system can be defined as a way for naming files in a computer [16]. It controls the storage and retrieval of files on a media (hard drive or CD or flash drive). It acts as an index containing the physical location where actually the data is present in a media. Every operating system whether it is Windows or Macintosh or UNIX, each includes a file system for storing files.

2.1.1 File System in Windows

The two main file systems in windows are:

1) Fat Allocation Table: File allocation table (FAT) is a file system which was developed by Microsoft in 1977 [11].It was the primary file system from MS-DOS to Windows ME. From Windows NT, NTFS became the primary file system. Earlier FAT was designed for floppy disks but later it was universally used for hard disks. The various versions of FAT include- FAT8, FAT12, FAT16, and FAT32. It is still considered as a preferred file system for floppy disks and high capacity storage devices like flash drives.

2) New Technology File System: It is also known as NTFS. It was introduced in 1993 by Microsoft in Windows NT 3.1. Starting from Windows NT 3.1, NTFS has now become the primary file system in all versions of Windows including Windows XP, Windows Vista, and Windows 7 and now in Windows 8 [11]. NTFS was developed with some improvements over existing FAT system. These are:

- Increase in security through file encryption and permission modes given to file user by file owner.
- It can support hard disk of larger size.
- Automatic recovery from some errors related to disks.
- Includes file compression for saving disk space.

2.2 Architecture of NTFS

In New Technology file system (NTFS), every object is considered as a file. This is because all the properties of files are inherited by the objects [17]. These properties

also include metadata of the file system which determines the architecture of the file system. In NTFS the architecture is divided into four units on a disk as shown below:

Fig 2.1: NTFS architecture.

Master File Table (MFT) is the most important feature of NTFS [18, 19]. MFT is divided into an array. The array includes records. Every file or every directory in the system needs to make an entry in the master file table. This entry forms a record in the array known as file record. The default size of record is 1024 B. Among the 1024 B, 42 B are used to store header of MFT and rest for storing attributes as shown below:

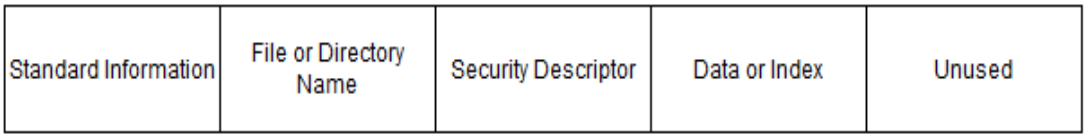

Fig 2.2: MFT record for a small file.

As every file is stored in master file table, so each data record contains attributes of the file. The common attributes are $DATA, $FILE_NAME etc. The description of all attributes is given in Table 1. There are two attributes that are present in every file, irrespective of the file type. These are $FILE_NAME and $STANDARD_INFORMATION. But there are also some attributes that depends on type of file. One among them is $DATA attribute which is present in case of files for storing file data and another attribute is $INDEX_ROOT which is present in case of directories for storing files of the directories. Besides these attributes, a file can have various other attributes that are not required by the Operating system [17].The attributes of a file can be:

- *Resident attributes*: resident attributes are those that store their contents into MFT table directly.
- *Non-Resident attributes:* non-resident attributes are those that store their contents in some external clusters. Then the name of the clusters used for storing the contents of that attribute is stored in MFT table as cluster run.

The various attributes of the files described in NTFS file system are explained in table below [19, 20, 21]:

Table 2.1: Description of various attributes in NTFS.

Serial No.	Attributes name	Description of attributes
1	$Standard Information	It includes information about total link count and also information about the creation, last accessed time , file modification date and access rights etc
2	$Attribute List	It includes information about the position of all the attributes required by a file but because of low space in MFT, it can't be stored in MFT Table
3	$File Name	It stores information about the name of the file. It stores both long and short file names. The length of long file names can be 255 characters (Unicode).The length of short file name can be 8.3 and are case-insensitive.
4	$Security Descriptor	It is required for determining the access to the file and also describes about the owner of the file. This tells who can have the access of file and who cannot.
5	$Data	Actual data of the file is stored in $DATA. In NTFS there can be multiple data attributes per file. By default every file in system has one unnamed $DATA attribute but it can also have one or more named $DATA attributes. Each name can have a particular syntax.
6	$Object ID	It is a unique identifier for a file. All files do not have object identifiers. Object ID is mainly used by the distributed link tracking service. Its size is 16 byte.
7	$ Logged Utility Stream	It is basically used by Encrypted File System (EFS). It is same as data stream but the only difference is that all the operations get stored to NTFS log file.

8	$ Reparse Point	This is used by installable file systems (IFS) for marking some files that are needed by that particular driver. It is used for volume mount points.
9	$ Index Root	It is used for implementation of indexes and folders. It is present in every directory and represents the root of B-tree of the index.
10	$ Index Allocation	Like $Index Root, $ Index Allocation is also used for implementation of indexes and folders.
11	$ Bitmap	It is used for implementation of indexes and folders. It is present in every directory.
12	$ Volume Information	It is used for storing version of the volume.
13	$ Volume Name	It is used for storing version of the volume.

2.3 NTFS Specific Data Hiding Method

There are various methods through which data can be hidden in NTFS file system but only those methods are considered appropriate that meet the following criteria with respect to security and capacity of data [17]:

- Standard utilities that are used for file system checking like chkdsk should not detect the presence of hidden data.
- The method should support hiding reasonable amount of data.
- Data which has been hidden should not get exposed by using standard graphical interface of file system.
- The method should avoid overwriting of the hidden data that means hidden data should be protected.

So only that method should be used that meets all the above requirements. If all the above criteria's are not fulfilled then that hiding method will not be successful in long term. If integrity of the data is not maintained that is data gets overwritten, then it will be difficult to retrieve data by forensic examiner or the person who has hidden the data. Also if the hidden data gets detected by standard utilities by common users then purpose of hiding data is of no use.

As mentioned above in NTFS, every object is considered as file, this means that an object can possess all the attributes of the file. Among the various attributes, $DATA,

$INDEX_ALLOCATION and $INDEX_ROOT are given unique names because they appear multiple times [19]. In NTFS certain attributes are required depending on the file type but there is an important feature of NTFS, that, it allows file to have those attributes also that are not required by the system. This feature of NTFS can be used for hiding data without affecting system functionality.

Each attribute in NTFS is uniquely identified by an identifier which is of numeric types, and all attributes are always placed on the basis of their numeric values. In NTFS every attribute stored has specific formats according to which they perform their specific purpose. That means if any modification is done to these attributes the system will get affected. Thus they cannot be used for hiding purpose. There is only one attribute known as $DATA which has no specific format. This means if any changes are performed on this, it will not affect the system. So $DATA attribute can be used for hiding files in the system. This attribute can have any size.

There are there methods of hiding data in $DATA attribute. These are [17]:

- Hiding data by adding additional clusters in unnamed $DATA attribute.
- Hiding data using $DATA attribute in files.
- Hiding data using $DATA attribute in directories.

1) Data Hiding In Added Clusters: This method of hiding is based on using additional clusters than those allocated to an NTFS file and hiding data in them.

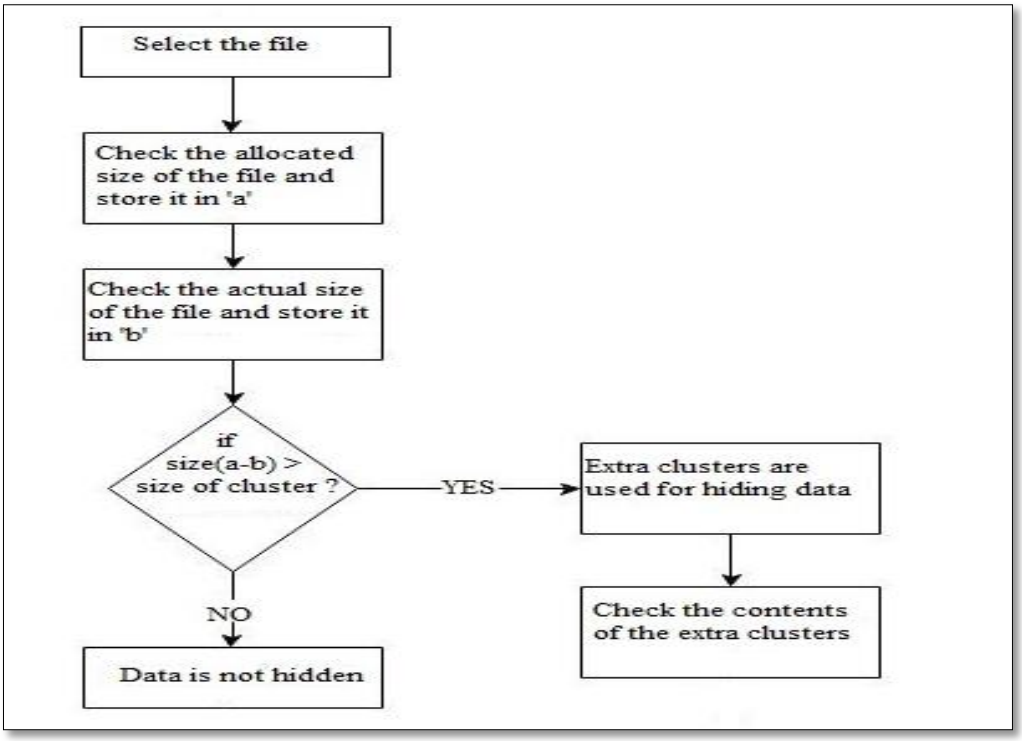

Fig 2.3: Analysis of data hidden in added clusters.

Suppose some clusters are provided by NTFS to a file for storing data then user can manually allocate some additional clusters to that file and can hide data in those clusters. Like others ways, here also the amount of data that a user can hide depends upon the size of the underlying NTFS file system as a user can allocate as many additional cluster for hiding data. There is one problem associated with this method, if the size of the original file starts increasing then it can overwrite the data hidden in additional clusters. So to prevent this only those files should be use for hiding data that are stable that is that will not increase their size in future. This can be prevented easily as the user who is hiding data by allocating additional clusters is the owner of that file and can deny write access to that file to other users.

2) Data Hiding in Files: As explained above, for each file there is an entry in master file table so an unnamed $DATA attribute is associated with each file but if a file record contains additional $DATA attribute other than the main attribute then the additional named $DATA attributes is alternate data streams [17].

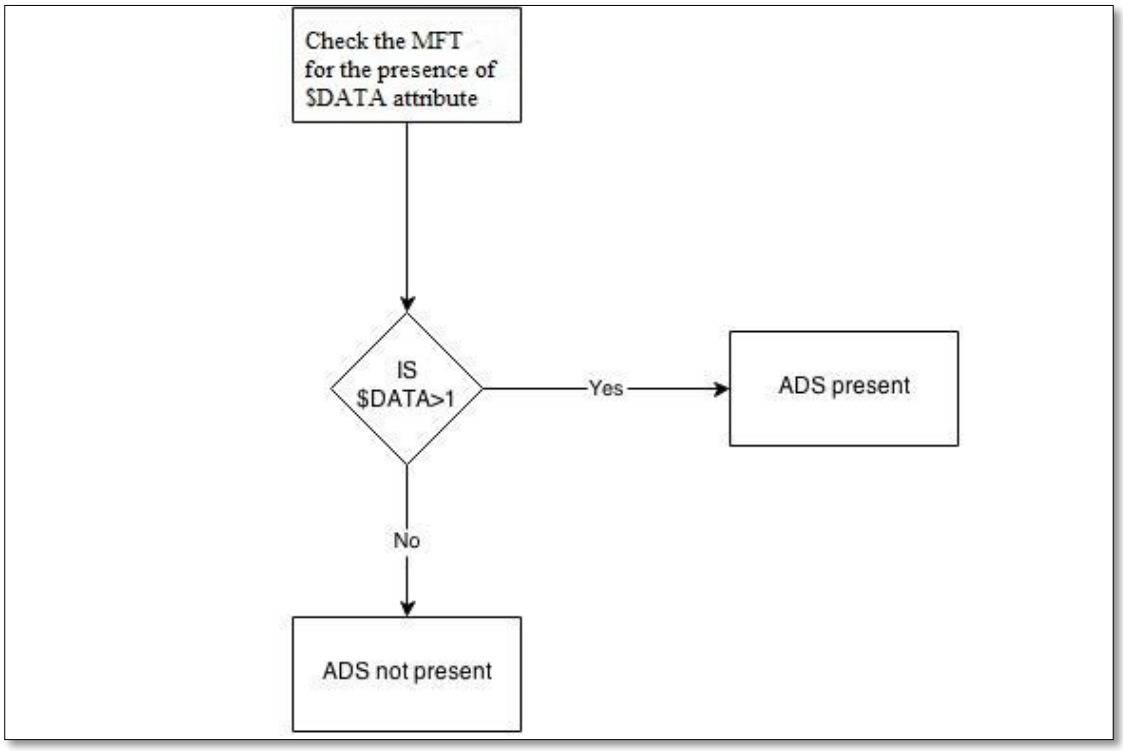

Fig 2.4: Analysis of hidden data in files.

In NTFS, ADS are added to make NTFS compatible with Macintosh file system and sometimes to provide additional details of some folders or documents [17]. ADS are used mostly to hide data because almost all the system utilities only check for first $DATA attribute which is of existing innocent file. So hidden files remain undetected.

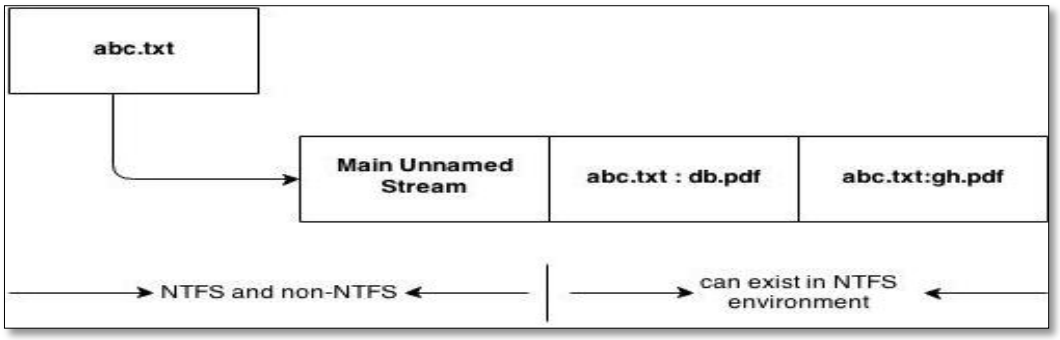

Fig 2.5: Visibility of ADS in NTFS and non-NTFS environments [22].

Figure above shows that the main file abc.txt is present in NTFS and non-NTFS area but if alternate data streams are attached to abc.txt then it is appears only in NTFS area not in non-NTFS area as non-NTFS drives do not provide this feature [22].

3) Data hiding in directory: $DATA attribute in master file table is used for storing information about contents of a file. So it is a common attribute for files but not in case of directories. In case of directories, $INDEX_ROOT is used for storing information as $DATA is used in case of files. So $DATA attribute is not necessary in case of directories. But it have been noticed that if validation checking is done using chkdsk for a directory, then even if $DATA attribute is present there, no error is shown. So this means that this $DATA attribute can be used for hiding data in case of directories also.

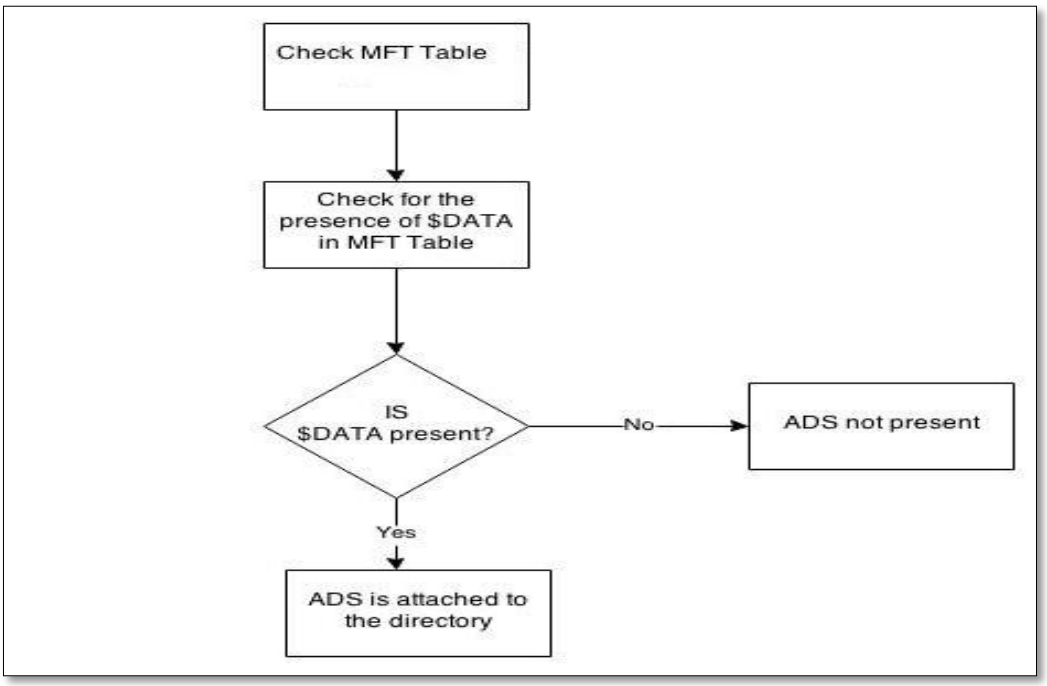

Fig 2.6: Analysis of hidden data in directory.

Thus alternate data streams can be created with directories for hiding data in the same way as in case of files. The size of data that can be hidden behind a directory depends on the size of underlying NTFS file system. The way of hiding data in directories is same as that of hiding in files.

2.4 Alternate Data Streams

Alternate Data Streams also known as ADS is a feature of NTFS file system in Windows Operating system [23]. It allows hiding data behind any legitimate system file [24]. Normally it is used for storing information about a file (metadata) but it can be used to hide a secret or malicious file inside the file record of an existing innocent system file. That is, when Operating system (windows) shows you a file, say "abc.txt", the metadata that tells your system from where to get "abc.txt" may also contain information about “EvilS.exe”. This means malicious files may be present in your system and you cannot see them using normal means. So it has become a vulnerability that can be easily exploited by hackers by hiding malicious files in victim machines.

It has been found that [25]:

- One can hide ADS behind any existing system file.
- One can add multiple files behind a single file.
- ADS are not affected by moving or copying file.
- The existing file is unaffected by the ADS.

2.4.1 Features of Alternate Data Streams

- Alternate data streams in NTFS are used to store additional information about a folder or a file. This additional information can be keywords, thumbnails, information related to program source etc.
- It can also be used for attaching any kind of file (executable files) to a file or folder.
- Can be used for hiding data as system utilities check only unnamed $DATA stream and so ADS remains undetected [26].
- Whenever directory listing is performed, ADS are not shown and even the size of the original file or folder does not change by adding ADS to it.

- ADS hiding technique in NTFS is much simpler than any other hiding technique as other techniques require use of low level file manipulation programs for hiding data like hex editor.
- The size of the data that can be hidden behind ADS depends on the underlying media used (NTFS size).
- While copying file having alternate data stream attached to it to USB drive, flash drive or non-NTFS drive then only main stream gets transferred and alternate stream is ignored.
- Internet Protocols like SMTP, FTP etc does not support alternate data stream. It implies alternate data streams cannot be exchanged over Internet.
- Files containing alternate data streams can be exchanged over local area network if the receiving system has NTFS file system.
- Various BHO (Browser helper objects) actually store malicious files in alternate data stream. The malicious files stored by them are detected by very few anti-spyware and anti-malware.
- A user can attach any number of streams to a file. There is no limit to it. Alternate data streams can be of any size and their size is not displayed by Windows.
- ADS are commonly used by hackers for hiding tools in a compromised system which can be later used for penetrating victim's machine [27].
- Malicious software can be easily hidden behind normal system file using ADS and remain undetectable.

2.4.2 Pros and Cons of ADS

Pros:

- Can be used for hiding data hidden through Steganography.
- Allows compatibility of Window's NTFS with Macintosh's Hierarchical File system.
- It is used by NTFS file system for storing information about a file like thumbnails (includes information about an image file) are hidden as ADS behind the main file.

Cons:

- As alternate data streams cannot be detected by normal system utilities so it has become a serious problem as it provides a perfect way for hiding data.

- Another major issue with alternate data stream is that once malicious files are hidden, these files can start hiding more and more data behind single file thus covering more and more disk space and causing denial of service attack.
- Anti-virus programs have difficulty in detecting ADS.
- File wiping utilities have problems in removing ADS.
- Disk de-fragmentation and file integrity software cannot detect hidden streams.

2.5 NTFS Streams

In NTFS, for each file in system, there is present unnamed stream in the MFT table. This unnamed stream is normal and viewable and data is present in this stream. The full name of stream is given as [28]

Filename: stream name: stream type

As in case of normal file, the default stream is unnamed so in that case the full name is

Filename: stream type

If file name is abc.txt and stream type is $DATA so full name is

abc.txt: $DATA

When an additional stream is attached to that file then a named stream gets added to it. This additional stream is known as alternate data stream. In this case full name of stream is

abc.txt: $DATA: $DATA

Here first $DATA is stream type of main stream and second $DATA is stream type of attached alternate stream. Suppose stream named dfg.txt is attached to main file abc.txt the full name is

abc.txt: dfg.txt:$DATA

If someone creates an alternate stream to a file which does not exist in the system then in that case automatically an unnamed stream is created in MFT table of zero length. So when the named stream that is the system file is deleted then all the attached named streams, that is alternate data streams also get deleted. This means when instruction for the deletion of file is given then security descriptor and all attributes along with unnamed stream gets deleted.

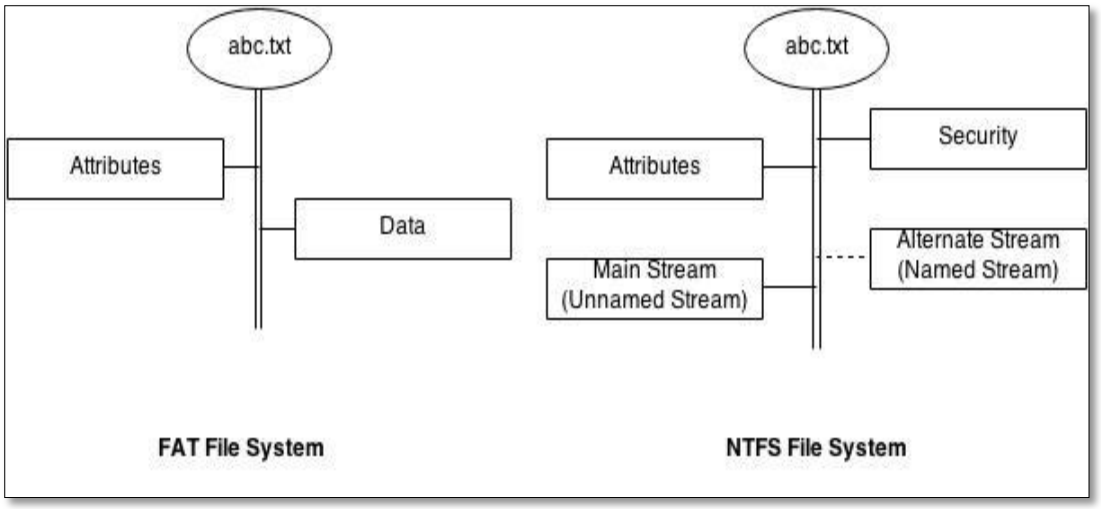

Fig 2.7: Comparing architecture of FAT and NTFS.

In figure it is shown, in case of FAT, for each file there is associated only attributes and data but in case of NTFS for each file, there is associated unnamed stream, attributes and security information or in some cases named alternate stream.

When a normal system file is opened then it implies unnamed stream is opened. In order to open alternate data stream, unnamed stream (filename) is appended by colon and named stream (alternate stream name) as

File_name: alternate_stream_name

An alternate data stream in case of directories is accessed in same way as in case of files. But the only difference is in case of directories there is no unnamed stream. In case of directories the default stream name is $130 and stream type is $INDEX_ALLOCATION [28].

2.6 Hierarchical File System

Hierarchical File System (HFS) was introduced by Apple Computers in 1985. Apple Inc. made it proprietary file system for computers running Mac OS. HFS was initially designed for floppy and hard disks, but now it can also be found on CD-ROMs. Apple introduced HFS to replace Macintosh File System (MFS). It has now become Mac OS Standard. HFS replaced MFS because searching the disk was slower on MFS in case of larger disks. HFS can support larger disks then MFS as it uses 32 bit integers for addressing [29].

Since HFS has become the default file system of Mac operating system so it is used for organizing files on hard disk of Mac operating system. It creates directories in hard disk. When new files and folders are added, the size of the directory expands

automatically. As in Windows the default file system is NTFS so Windows cannot recognize the disks having HFS as their file system. So in order to make Windows and HFS compatible with each other the feature Alternate data streams is added in NTFS.

HFS supports maximum volume size 2TB and maximum file size 2 GB. In HFS file name length can be maximum 255 characters. B-Tree is used in HFS for managing directories and their contents [29]. The file system in HFS is case insensitive. The file names and directory names are stored with their case intact.

In HFS each file includes:

- **Data Fork:** It stores information about the data stored in the file.
- **Resource Fork:** It stores information about the file (metadata of the file).

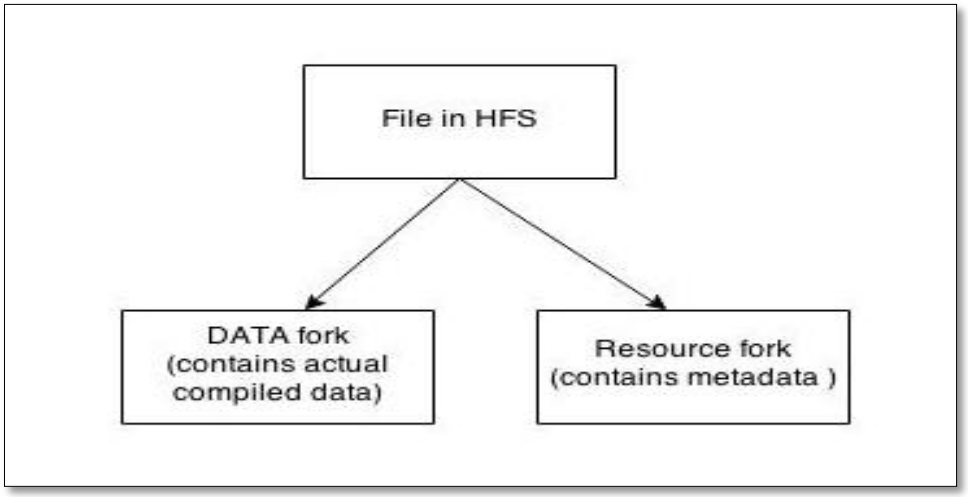

Fig 2.8: Architecture of file in HFS.

2.7 Requirement of Alternate Data Streams

In Windows concept of Alternate Data Streams was introduced by Microsoft for supporting Macintosh files [27]. As discussed above a file stored in Hierarchical file system of Macintosh Operating system includes both- data fork and resource fork. As explained data fork contains the actual data whereas resource fork contains information related to the file like what kind of file it is, which application is needed to open or create that file etc. Resource fork in Macintosh is similar to extension in Windows. Both are used to find the correct application or software for opening the file [30]. So in order to make Macintosh Compatible with Windows [26], the resource fork of Macintosh is stored as alternate data stream in Windows which is attached to the main file.

In spite of storing only information about file type, ADS includes a lot more information in the same way as resource fork in Mac do. This is the reason why Alternate data streams are included in a number of Windows programs. One of the most common examples of ADS is Microsoft Word. In case of Microsoft Word information describing document that is annotations are stored in ADS which is attached to the document [31]. There are various other legitimate uses of alternate data streams that are explained later.

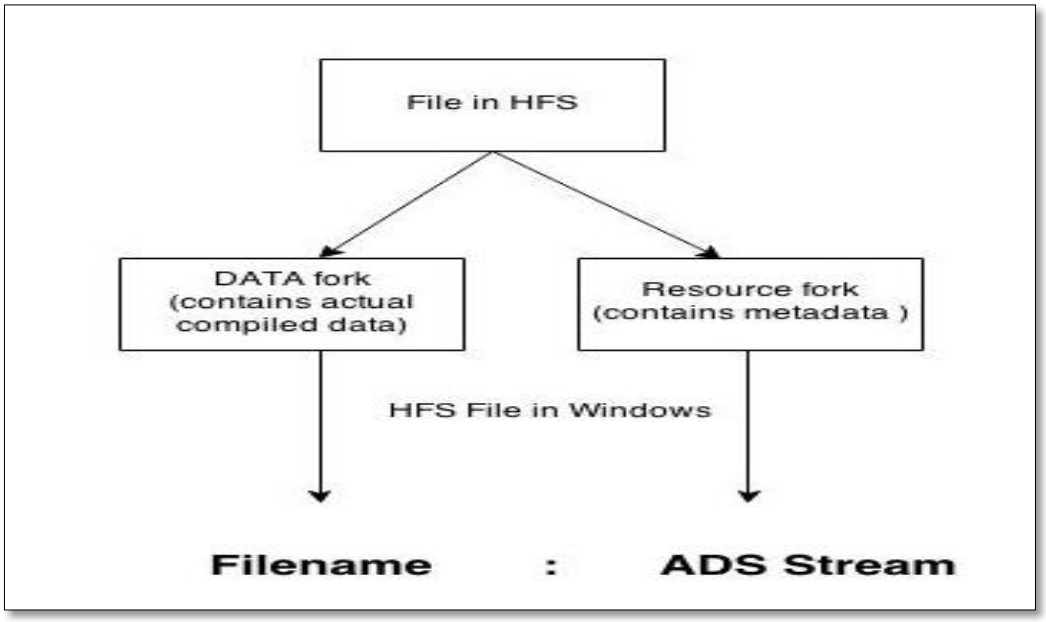

Fig 2.9: Compatibility of Windows and HFS.

2.8 Detecting ADS in Forensic Investigation

Major part of Computer Forensics is data hiding. Computer Forensics deals with solving many cases related to hidden data [32]. Data is mainly hidden so as to perform malware attack on a computer system. This hidden data is then used to compromise victim's system. Sometime criminals try to hide their criminal activities in computers [33].

Various methods are available for hiding data. Among them one method is an alternate data stream which is found in NTFS only. When an attacker gets access of a system then attacker hides some files in victim system. If an employee tries to do some unauthorised attempts on system then also employee makes some files invisible from third party so that they cannot know about it. NTFS streams are used for both above cases [34].

2.8.1 Forensic Challenges

The main challenge of forensic investigators is to prevent the use of alternate data streams for malicious purpose. As these streams are hidden so it becomes even more difficult to locate them. Software are needed for searching them in NTFS drives. These streams can be present in any NTFS partition whether it is zip disk, Email attachment, hard disk etc. These can be moved from one NTFS system to other. It is also true that if the NTFS data is moved to non-NTFS area then these streams are lost. This can be seen as both an advantage and disadvantage. Advantage as this will remove alternate data streams permanently. The disadvantage is that when backup is made of NTFS drive to non-NTFS drives like CD-ROM, Jazz drive etc then alternate data streams are not backed up. So this will create problem for forensic department as they will not get complete backup. If backups are not complete then forensic investigators will never get to know about the real cause of system compromise. The various challenge that computer forensic are facing are:

1) Backup software: The backup software available for Windows system having NTFS as their file system does not provide full support for alternate data streams. They provide only partial support to ADS. Computer Forensic tools like lads, streams etc can detect the presence of alternate data streams in the system but they cannot detect them in backups of file systems [22]. Backup software available; destroy alternate data streams while restoring data. Forensic investigators believe backup software to be such that:

- If contents of a file are backed up to NTFS file system then it should be able to restore to both NTFS disk and FAT disk.
- If contents of a file are backed up to FAT file system then it should be able to restore to both NTFS disk and FAT disk.

But it has been found that, no software is capable of retaining all the data contained in alternate data stream when backup is created and also no software is capable of recreating whole data of alternate data streams when backup is restored. So computer forensics should find a way in which no loss of data of alternate data stream occurs. It has been found that-

- 50% of the backup software does not consider alternate data streams while backup.
- 30% of software backup alternate data stream in NTFS only.

- 20% of software available that can backup alternate data streams properly and keep them intact in NTFS and non-NTFS file system. Also they can restore alternate data streams properly in NTFS area.
- No software is available that can completely backup and restore alternate data stream in any platform whether it is NTFS or non-NTFS area.

Normal user considers backup as process of saving data and later restoring selected data. But for computer forensic, it's very important as unless data is recovered, it cannot be verified and the main task of forensic is to check the contents of the data. Forensic experts do not want to lose the contents of alternate data streams. So once data is restored from backup, the forensic experts scan that data using alternate data streams detection tools so as to know whether ADS are present or not. If ADS are present then next step is to extract or fetch them. If after backup, restoration takes place at non-NTFS area then ADS are lost completely. So that software is needed that can restore data having alternate data stream in non-NTFS area. In this case all ADS related data will get stored in specified folder. So the need of software for detecting alternate data streams is no longer required.

2) Antivirus: Antivirus monitors are software that are used for checking files for the presence of virus and other malicious files. This software works as background process to monitor all the operations carried out on files like opening and closing of files [35]. The antivirus monitors are basically of two types:

- Some monitors check files of the system without paying attention to file name and extension.
- Other type of monitors mainly considers file names and extensions. Thus they consider only primary data stream not alternate data stream. So they are of no use to forensic investigators.

As monitors require much system resources [26], so administrators do not prefer to use them on servers. They prefer using antivirus scanners that do not require much system resources. Scanners do not affect system speed as monitors do. The problem with antivirus scanners is that they do not check for the presence of alternate data streams [26] because vendors of antivirus scanners believe that even though virus are hidden in alternate data stream, still attacker needs to install a starter in main stream so as to invoke the malicious file residing in alternate data stream. So if starter is detected by antivirus then presence of malicious file can be detected so there is no

need of checking the alternate data stream. But there are problems in this theory of antivirus vendors:

- Even though these antivirus check the starter for knowing about the presence of malicious data in alternate stream but still they cannot affect alternate files. They cannot harm them.
- There are various ways available for invoking files present in alternate data stream without even affecting the main stream. So scanners will not get to know about the presence of infected data in hidden stream.
- Virus starters give random names to the main streams in which they are present so detecting them becomes very difficult.

So detecting for the presence of malicious files in alternate streams becomes very difficult.

3) Compressed Alternate data stream: Another challenge for forensic experts is compressed alternate data streams as there is very less knowledge about this. Detection of alternate data streams that are compressed is even more difficult than that of detecting normal alternate data streams. There is no proper procedure defined by forensic experts for detecting alternate data streams in compressed form.

For compressing an alternate data stream, compression software like WinRAR is needed. For understanding, consider a file abc.txt having 0 bytes of data, attach an alternate data stream of size 26 bytes to abc.txt. This can be done by opening command line, creating an alternate data stream ads.txt and storing content “this is alternate data stream” to it and attach it to abc.txt.

echo “this is alternate data stream” > abc.txt:ads.txt

Now right click on abc.txt and select add to archive then following dialog box will appear. In the advanced options, select save file streams. This will include alternate data streams attached to abc.txt

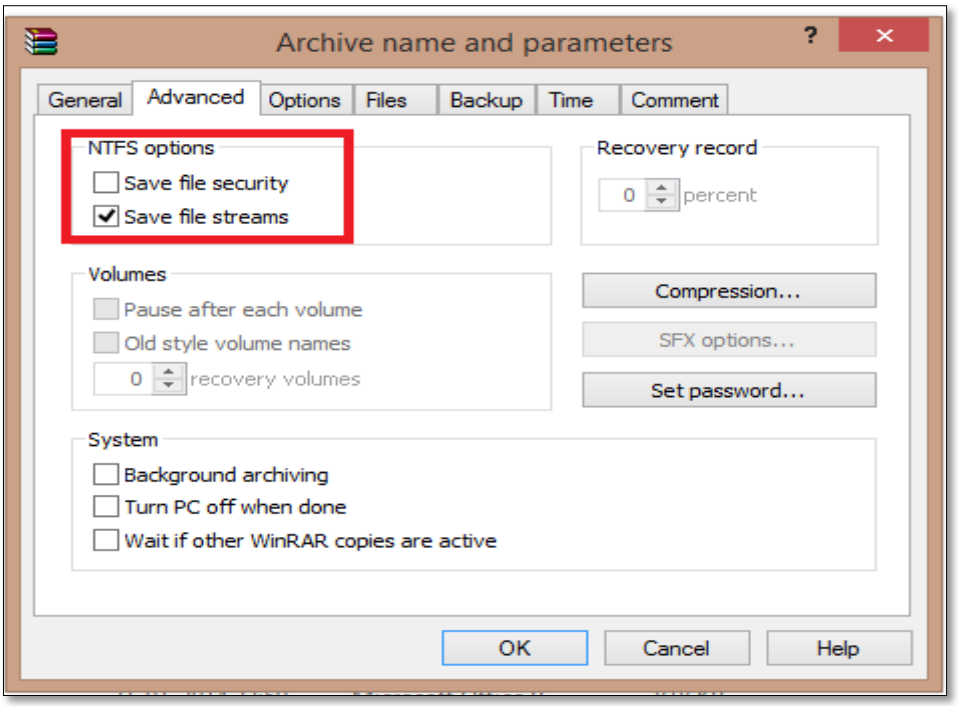

Fig 2.10: Showing saving ADS while compressing a file.

After selecting ok, compressed alternate data stream is created. After decompressing, the size of abc.txt is 0 bytes but that compressed file abc.rar is 50B bytes. From this one can conclude that along with normal file, some other file is also present. If the compressed file is decompressed on non-NTFS area then only main file will exist and ADS will destroy [32].

Once alternate data streams are compressed then

- Transfer of alternate data streams over internet become possible.
- In compressed form alternate data stream can be stored on non-NTFS volume.

Malicious use of compressed alternate data stream can be explained with the help of example suppose a user installs software which contains a virus by mistake. Once the file containing virus will detect the existence of NTFS drive in system, then that virus will copy all the compressed files that contain backdoor in alternate data stream. After decompressing that file, the backdoor will execute and will collect all the information about user's system. If a malicious file is hidden in an alternate data stream and that file is compressed then detection rate decreases.

2.9 Legitimate Use of ADS

Alternate data stream was included by Windows in its NT version. There are various legitimate uses of alternate data streams [36]. These are:

1) Services for Macintosh: These were introduced in NT 3.1 having NTFS as file system so as to make Windows NT as file server for clients using Macintosh Operating system. In hierarchical file system, that Mac uses, data is stored in data fork and information about that data (metadata) is stored in resource fork. So as to store the Mac files, Window NT, stores the data in its primary main stream and metadata of that file in alternate data streams. This makes the Windows and Mac compatible with each other

2) Summary Data: Microsoft introduced summary tab from Windows 2000, to include the properties of some selected files. The summary tab includes information like file title, author of the file and subject. This summary information was stored in alternate data streams attached to selected file. In case of images, thumbnail that includes metadata about an image are stored in ADS.

3) Volume change Tracking: It means monitoring changes that are made to a NTFS volume and then taking actions if required. These changes can be checked by looking at the change journals rather than looking for modifications done to the entire files. These change journals are stored as hidden streams (alternate data streams). It is mainly needed by backup applications, scanners etc.

4) Zone Identifiers: When a file is downloaded from internet in an NTFS drive, then an alternate data stream gets attached to the downloaded file. These alternate streams (zone identifiers) store security information about a file. This stores information about the source from where the file is downloaded [37]. These are stored as:
Filename downloaded: Zone. Identifier
The content of this Zone.Identifier is as shown in figure in a notepad file is:

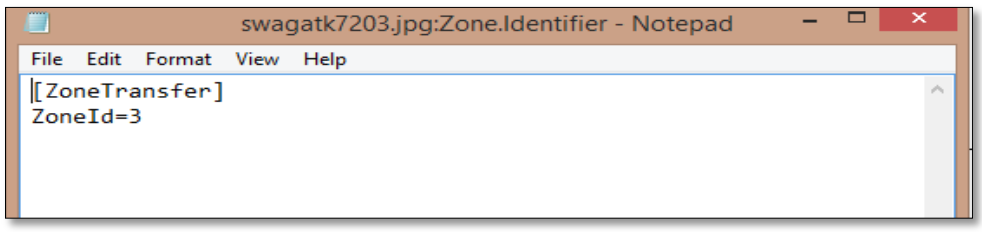

Fig 2.11: Contents of file containing Zone.Identifier.

Fig 2.12: Dialog box asking for user permission for executing a file.

When a file having Zone.Identifier in alternate data stream is executed then following warning comes. If the user uncheck this 'always ask before opening this file' then this dialog box will not appear and ADS will be deleted. This is mainly for security purpose only. By right clicking the file and opening properties dialog box, if the user selects unblock then also Ads will get deleted.

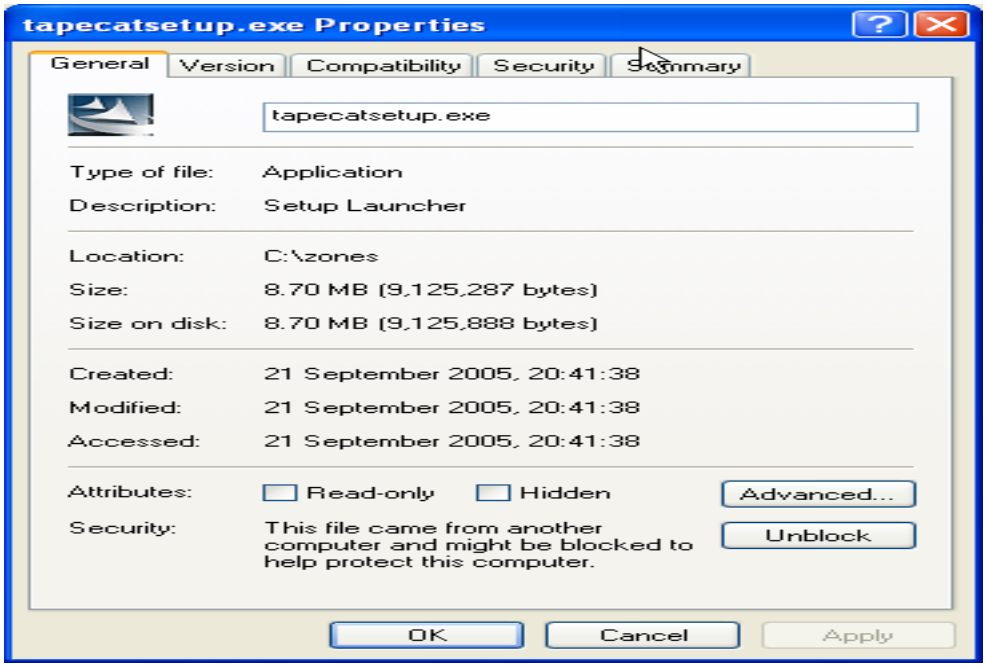

Fig 2.13: Showing capability of deleting Zone.Identifier attached to a downloaded file.

2.10 Windows Applications that do not recognize ADS

There are three types of applications that do not appear to recognize alternate data streams. These tools do not behave properly when alternate data stream do appear in system [36].

1) Windows Tools: The default tools available with Microsoft operating system are considered to be aware of alternate data streams but actually they are not. Even the latest version of Windows that is Windows 8, do not that such tools that can recognize the hidden streams. The windows commands used for directory listing like dir only show the primary data stream. They do not show alternate data streams. But these tools like window command line provides user with the ability to create alternate data streams by copying data of primary data stream to alternate stream. Also notepad application of windows is capable of reading these alternate data streams. But notepad cannot detect the presence of alternate streams like other windows tools.

2) Virus-Scanners: Alternate data streams came into existence from Windows NT, About 14-15 yrs are past still anti-virus fail to recognize their presence in the system. These scanners only check the default stream for the presence of virus. Only the heuristic antivirus that check the behaviour can detect the presence of malicious files in real-time mode but as discussed they cannot recognize alternate stream so they cannot harm these streams that contain malicious code. So third-party tools that can scan the system for the presence of alternate data streams are needed.

3) File Integrity Software: This software maintain a database having a unique value for each file in a system. This unique value is calculated on the basis of the contents present in that file. The database once maintained by this software can be later used to verify whether any changes are made to the file or not. The various applications that are used for calculating the unique value are md5, CRC (cyclic redundancy checking) algorithm, etc.

From the studies, it has been found that the integrity software do not consider alternate data stream as the md5 value for a file without having alternate data stream attached to it is same even if it has an attached alternate data stream. This means they only protect the primary data streams but not alternate data streams.

2.11 ADS Vulnerability

It has not been cleared yet whether ADS is a threat or functionality. Although there are some advantages of hiding data in existing files still there are various disadvantages too. The major disadvantage is its ability to go unnoticed. So in order to better understand this there are four major areas to be discussed [38]:

1) Virus Attacks: Alternate data streams allow user to hide VBS, EXE, and Cmd or Bat files. This makes Alternate data streams a good source for hiding malicious files. So it becomes difficult for antivirus software vendors to deal with these streams. Virus scanners mainly scan the default data streams of the files. So in order to detect ADS antivirus software vendors point out that, real time scanners can detect malicious file at run time that is when that file gets executed. So this means ADS can be detected at run time. The only issue with this is that real time scanners slow down the speed of the system because of which various administrator do not use them on servers. So if real-time scanning will not allowed then it will not be possible to detect virus during normal scanning of file system. Example of a real virus attack which took the advantage of alternate data stream is:

W2K. Stream virus- It was created by Benny and Ratter in September 2000. This virus basically took the advantage of the NTFS alternate data streams [39]. In year 2000 this virus infected various files on Windows 2000 version. It infected only those files which were present in the same directory in which this virus was present.

The virus was 3628 bytes in size. It used to replicate only in those systems that had NTFS as their main file system. After replication this virus used to replace itself with the host file and stored the actual content of host file in a named stream (Alternate Data Stream). The size of host file was changed to 3628 bytes. The main problem that occurred due to this virus was that until the host file was in NTFS environment, its actual content remained saved in stream attached to it but as soon as the host file was moved to non-NTFS environment then its contents were destroyed.

2) Backups: After understanding the nature of ADS file system, it has been found that file system backup systems can only backup default streams of file. This means Backup of ADS files cannot be created. The main problem with this approach is that if a user change or remove any data in named streams that will not be noticed. The main reason behind this is that ADS exist in new technology file system. So if we move a backup of files to FAT disk, all the named streams attached to it will get

removed. Backup software vendors that provide support for backup of ADS are Veritas and Network Associates. But problem is that many users are still using the old backup software that does not support alternate data streams.

3) Denial of service (DOS): It is also used for exploiting ADS. Although DOS is a common attack still as ADS files are undetectable so this attack has further increased the threat. For example an attacker can attach large series of ADS to a file so that system partition gets filled. This will result in crash of server as there is no space left for storing temporary or paging files. When someone use default files for launching DOS attack, these files can be easily detected with the help of third party software that monitor size of the files by scanning a directory for abnormally large files. But if someone perform DOS with ADS files then it becomes difficult to detect. Another attack that can be performed with ADS is attaching more than 6000 ADS to a file. If attacker tries to access that default stream on which more than 6000 files are attached then system response becomes slow in best case and system does not response in worst case.

4) Data Hiding: ADS is mainly concerned with hiding data. The main threat with ADS is difficulty in detecting alternate streams attached to default streams. So if an attacker hides malicious code in a default stream then the size of the default stream will not change. So in order to detect there streams, network administrators should be aware of various tools available for detecting these streams.

5) Exploiting a web server: Alternate data streams can be used for exploiting a remote web server [40]. The source file of some web servers can be read using: $DATA stream. If those servers are using PHP or ASP as server side script languages which have some vulnerability and are not properly patched then in that case, the source code of the web server can be easily viewed by an attacker by using the URL as shown:

http://www.fhgh.com/index.php::$DATA

After getting the source code of a web server, various critical information, like information flow in web server, coding details etc can be known. Using this information, an attacker can attack the server.

Summary: In this section, various hiding schemes in NTFS were explained. Alternate data stream is one of them which is not much documented. It was introduced for making HFS compatible with NTFS and also for various others legitimate uses. There are various advantages and disadvantages of Alternate data stream. Windows have introduced this feature but still windows tools are not able to recognise it because of which it has become vulnerability and is easily exploited by hackers for attacking systems. This undetectable nature of Alternate data streams has thus made it a challenge for forensic analysts.

CHAPTER 3

PROBLEM STATEMENT

3.1 Gaps in Study

Data hiding using Alternate data streams is a very important feature of NTFS. This feature is not properly documented so awareness among computer users is very less regarding this. So due to lack of awareness, this feature has become a boom for intruders. They consider it as a vulnerability and use this as a proper way for hiding malicious files in victim's system as this is the only way of hiding which is totally undetectable. Only third-party software can detect the presence of Alternate data streams. So users need to be aware of this feature of NTFS. Users can use this feature for hiding their personal information that they want to hide from others. But as no software is available in market that can provide this hiding capability to normal users so this feature cannot be utilized. Users need to use command prompt for hiding files and for this they have to learn and remember various commands. This is quite difficult for a normal user. Also software that are available in market and are commonly known to people- Lads and Streams are command line. Among them, Lads only detect Alternate data stream but do not provide any provision for deleting them. Because of command line nature, many users have difficulty in using them. But no tool defines whether the ADS they are detecting contain malicious files or not.

3.2 Problem Statement

After studying the problems in the existing system, the conclusion drawn is that there is no third party software provides the capability of hiding data in Alternate data streams, so the first task is to develop a system that solves this problem by providing a graphical interface that enables the user to utilize this feature without the need of remembering the commands. The second problem is to develop an encoder that provides zero detection rates. So that hidden data remains unreadable even after getting detected. Also no software is available that predicts whether the detected alternate data stream contains malicious file or not so to solve this, there is a need of virus scanner. Sometimes unknowingly user deletes legitimate Alternate data stream so in that case also virus scanner will be very helpful. For solving all these problems, a system is proposed which not only provides hiding capabilities but also detecting

and deleting capabilities. Also as there is very less documentation about alternate data streams, so users are unaware of the vulnerable side of ADS. Thus last problem is to demonstrate the attack that can be performed on system by exploiting ADS.

3.3 Objectives

1. To develop a system capable of creating, detecting and deleting Alternate data streams.
2. To add stealth in the system by introducing a Fully Undetectable (FUD) encoder.
3. To compare the accuracy and performance of developed system with the previous systems available.
4. To demonstrate the attack that can be performed on a Windows machine by exploiting Alternate data streams.

CHAPTER 4

IMPLEMENTATION

4.1 Stealth ADS (SADS)

The proposed system illustrated here is an open-source GUI based system that performs following major tasks:

- Creation of ADS
- Detection of ADS
- Deletion of ADS
- Addition of stealth using inbuilt encoder
- Scanning for presence of malicious file.

4.1.1 Creation of ADS

With Stealth ADS, one can easily hide a file behind any legitimate existing system file. The users need not to remember all the complex commands for hiding and creating ADS, as in case of the normal command line system. The system makes creation of alternate data streams easier for any normal user not only for hackers. Using this system a normal user can hide personal data like passwords, ATM pin number etc that are difficult to remember. Using this system, the sensitive data remains protected in encoded form and no other person can read and manipulate it. Data is encoded with the help of encoder added in the system which is explained later. Similarly in companies the employees can secure their data from attackers by hiding that data in encoded form using SADS. So confidentiality and integrity of data is maintained. Thus SADS provides ease and fastens the ADS creation process. The various types of files that can be hidden with the help of SADS are:

- Text files
- Images
- Executables
- Videos

SADS also provide functionality of executing or running hidden files. The files can be launched either using default system programs or by selecting any other program for launching. The whole functionality is explained in the below flowchart:

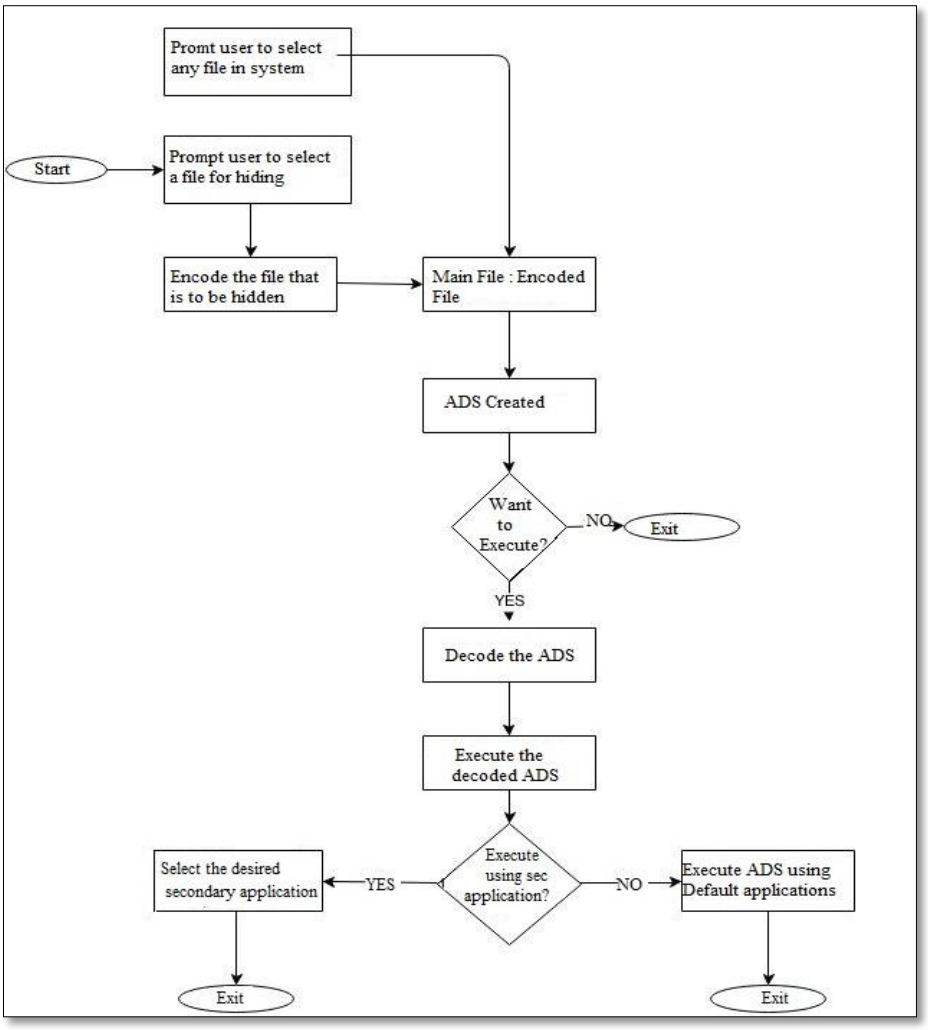

Fig 4.1: Flow chart showing the control flow of creation module.

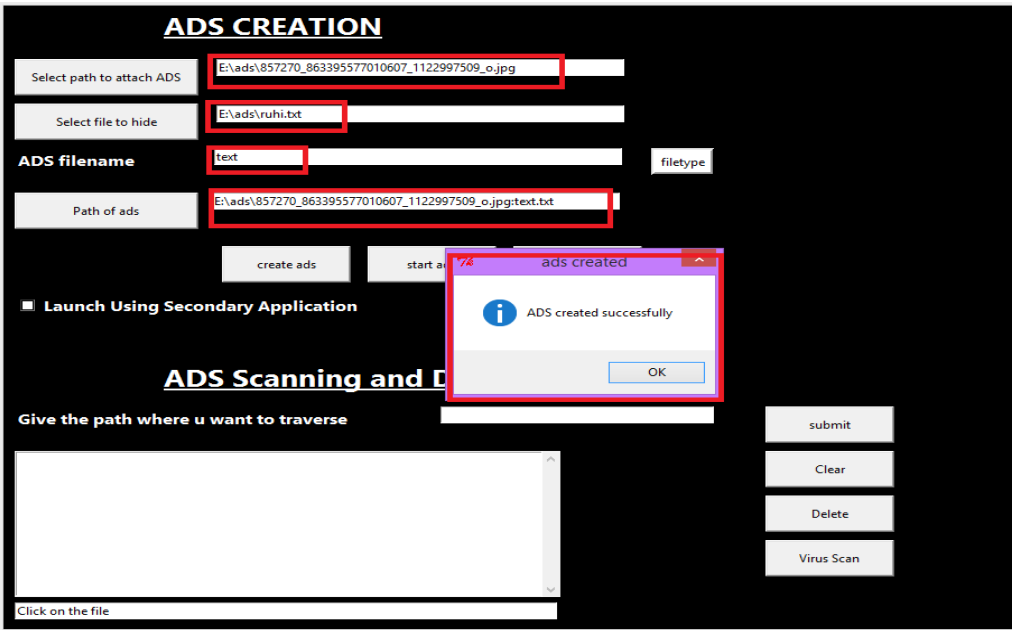

Fig 4.2: Stealth ADS showing hiding a text file behind an image.

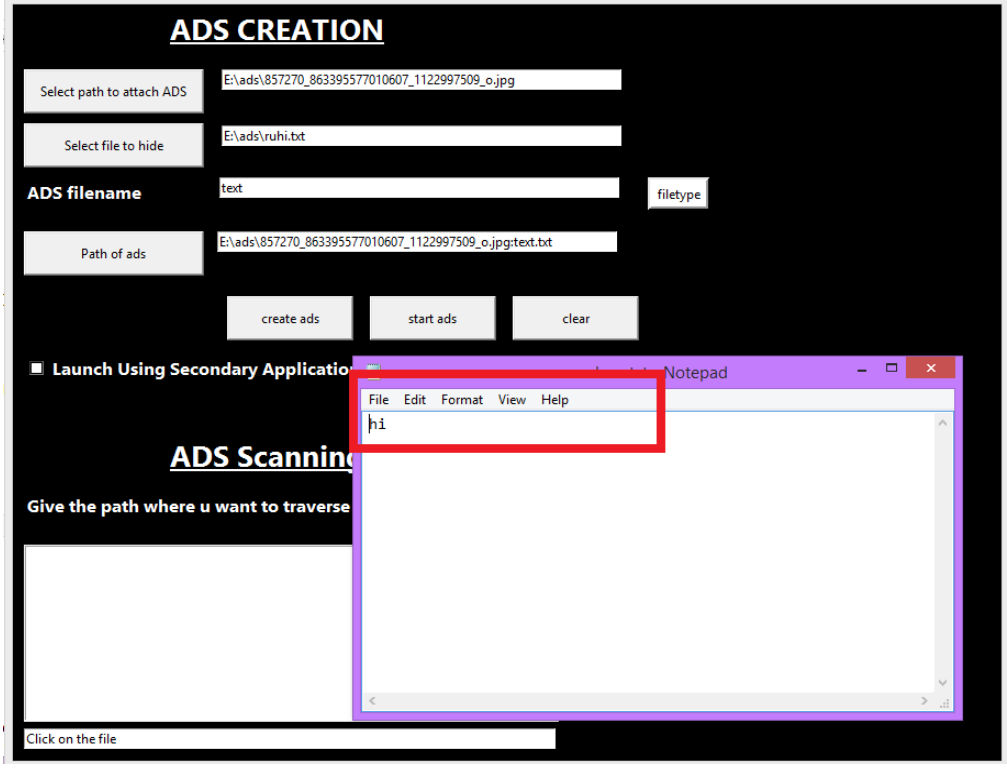

Fig 4.3: Opening hidden file text.txt.

4.1.2 Detection of ADS

Stealth ADS provides functionality for scanning system for the presence of alternate data streams. It is the preventive measure against the malicious use of alternate data streams. If an alternate data stream containing malicious code is present in the system then the user will never come to know about it unless system is checked for the presence of ADS.

Detection of Alternate Data Streams is quite difficult by using normal system utilities like ‘dir’. As the size of the file to which ADS are attached is not changed so it makes detection much difficult. SADS enables to check for the presence of ADS in the system. It provides the whole path of the file including the directory in which the alternate data stream is present. Once the files are detected the user can take important steps, whether files should be deleted or not.

Logic used: Detection process is based on checking the entry of the file in the MFT table for the presence of an additional $DATA stream. A normal unnamed stream is always present in MFT table for each legitimate file but if additional $DATA attribute is present then it means there is an alternate data stream attached to the file. The flow of detection process is shown in the figure:

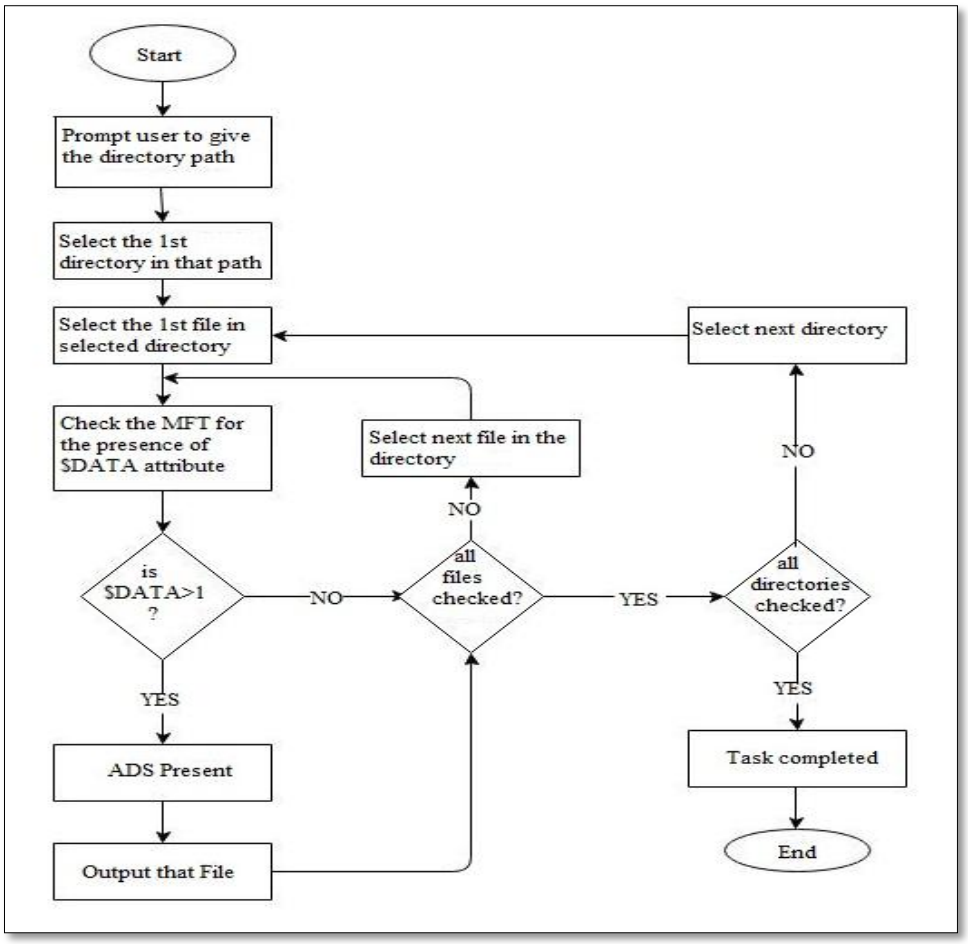

Fig 4.4: Flowchart showing control flow of detection module.

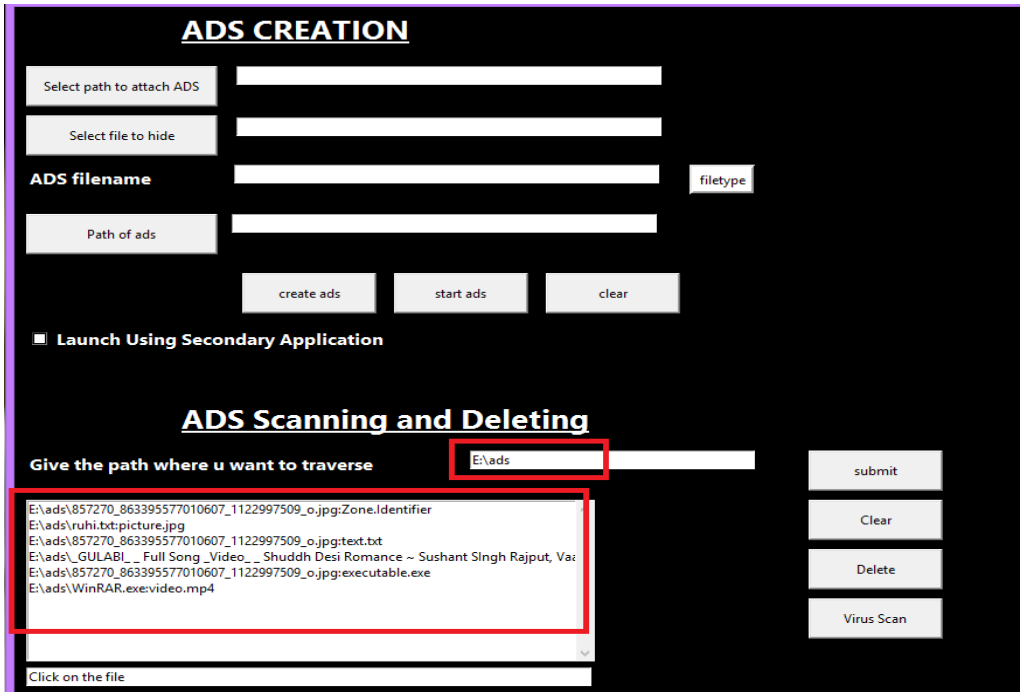

Fig 4.5: SADS showing all ADS present in directory E:\ads.

4.1.3 Deletion of ADS

After the detection of the presence of ADS, the next step is to get rid of unnecessary hidden files which are either not required or which are malicious in nature. While deleting the unnecessary ADS one has to make sure that the integrity and availability of the original file remain maintained. ADS cannot not be deleted by using the windows delete command directly as it will result in the deletion of the original file also. This method can be used only in cases when the original file is not required. But this method is not suggested.

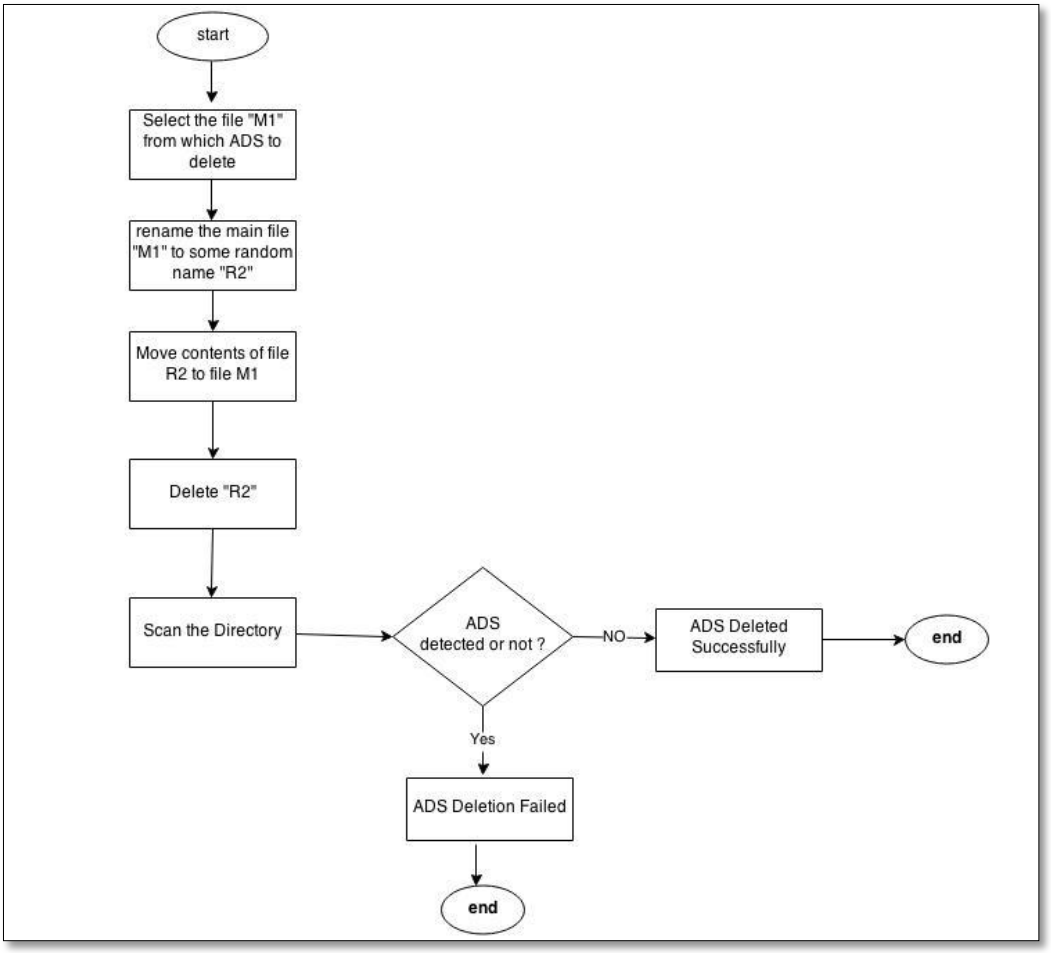

Fig 4.6: Flowchart showing control flow of deletion module.

Logic Used: The main logic used behind deletion in SADS is explained below:

ren filename temp.exe

type temp.exe > filename

del temp.exe

where filename is the name of the file with attached ADS. In above commands, the logic is to first rename the original file name to some random name (e.g. temp.exe)

and then move the content of temp.exe to a new file. The name of the new file should be same as that of original file. This will copy the content only not the ADS as type command of windows does not copy ADS and then finally temp.exe can be deleted. As ADS files are attached to temp.exe so they automatically get deleted along with the temporary file. So using this method the original existing files remain unaffected. The whole flow is shown in figure above.

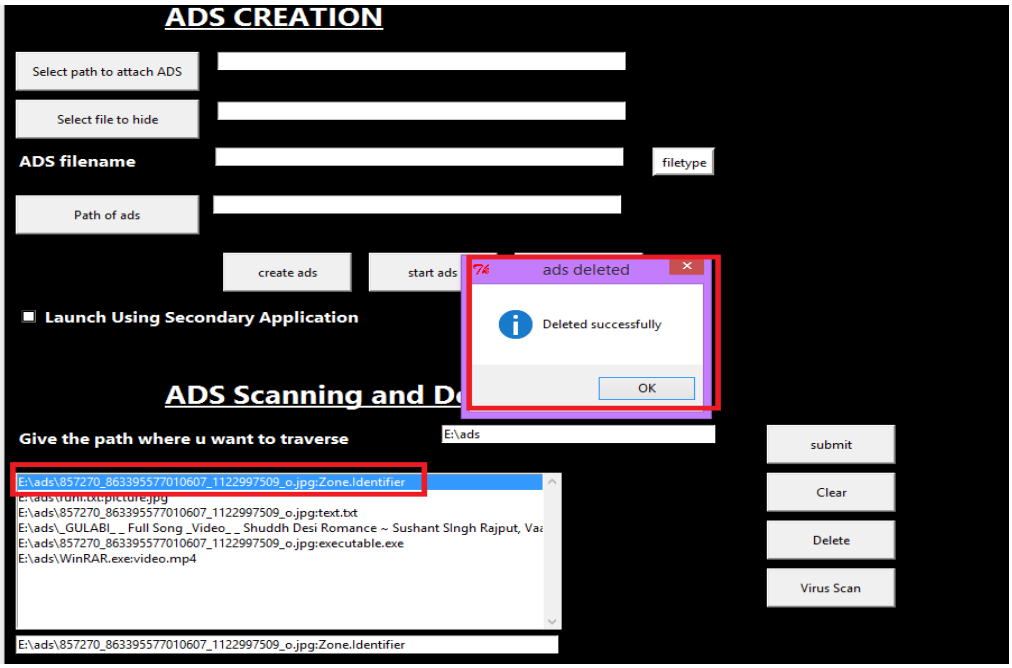

Fig 4.7: SADS showing deletion of selected ADS.

4.1.4 Encoder

The main feature of SADS is encoder. The encoder is included so as to add more stealth to the system. In SADS , before hiding a file, the file is encoded using the in-built encoder. It makes SADS more powerful as the hidden file is in encoded form and thus cannot be read in case it is detected by an unauthorised user. This means hidden file can be read only by the person who has hidden it. Thus using encoder sensitive information can be hidden properly in encoded form without the fear of detection. In SADS, before creating an ADS, the file to be hidden is send to encoder for converting it to unreadable form. Once encoded, the file is then attached as ADS. Later on, when ADS file needs to be executed then before opening it, the file is send to decoder for again converting it to original form. The working is shown below:

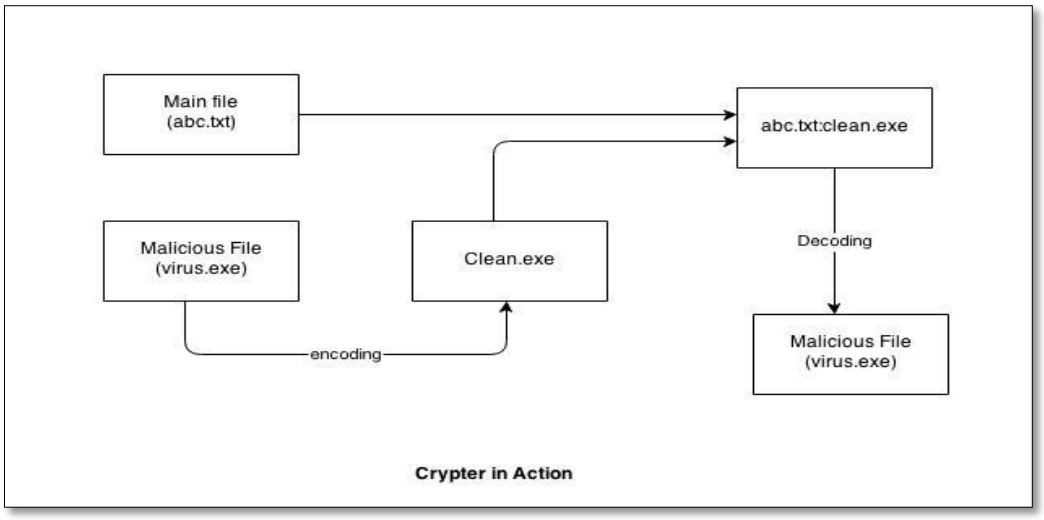

Fig 4.8: Encoding and decoding of an ADS file.

The encoder not only encodes the content of the file but it also changes the header of the file. Thus changing the headers to some random junk make it impossible for the antivirus to compute the hash and also make it impossible to determine what kind and format of file it is.

Logic Used: The logic implied in creating encoder in SADS is very simple. It makes use of hex encoding for crytping the files. The following two file handling modes are used while encoding and decoding the files:

1) Rb+ (Read Binary Mode): This mode allows opening any binary file in a read mode and thus allows reading and copying the contents of a binary file. Any binary files like exe, pdf, mp3, txt can be opened and supported by this mode.

2) Wb+ (Write Binary Mode): This mode creates the new file and opens it in a write binary file. This mode allows modifying the contents of a binary file. One can easily copy one binary to another via this mode.

Initially the file one wants to encode is opened in a rb+ (Read Binary) mode. So once the file is available in read binary mode, the next step is to read the content of the binary file and the contents are fed to the hex_encoder function which encodes the content into the hexadecimal format. Then a new file is created and opened in a wb+ mode and the encoded data is written to it.

Algorithm:

1. Select a binary file for encoding.

2. Open the selected file in rb+ mode.

3. Read all the contents of the binary file.

4. Until end of the binary file, send the content to hex_encoder function which will convert data to hexadecimal form.

5. Create and open a new file in wb+ mode.

6. Save the encoded hexadecimal contents in the newly created file opened in wb+ mode.

The new file thus generated will be the encoded file.

4.1.5 Virus Scanner

Scanner or more specifically antivirus scanner is another feature that is added to SADS. So SADS not only scan system for the presence of alternate data streams but also after detecting ADS it scans it to check whether it contains any malicious file or not.

As already discussed alternate data streams not only used for illegal purpose but also for legitimate use as for storing zone.Identifier, thumbnails etc so it is not a good habit to delete all alternate data stream without knowing what actually they contain. So this scanner will help in that case.

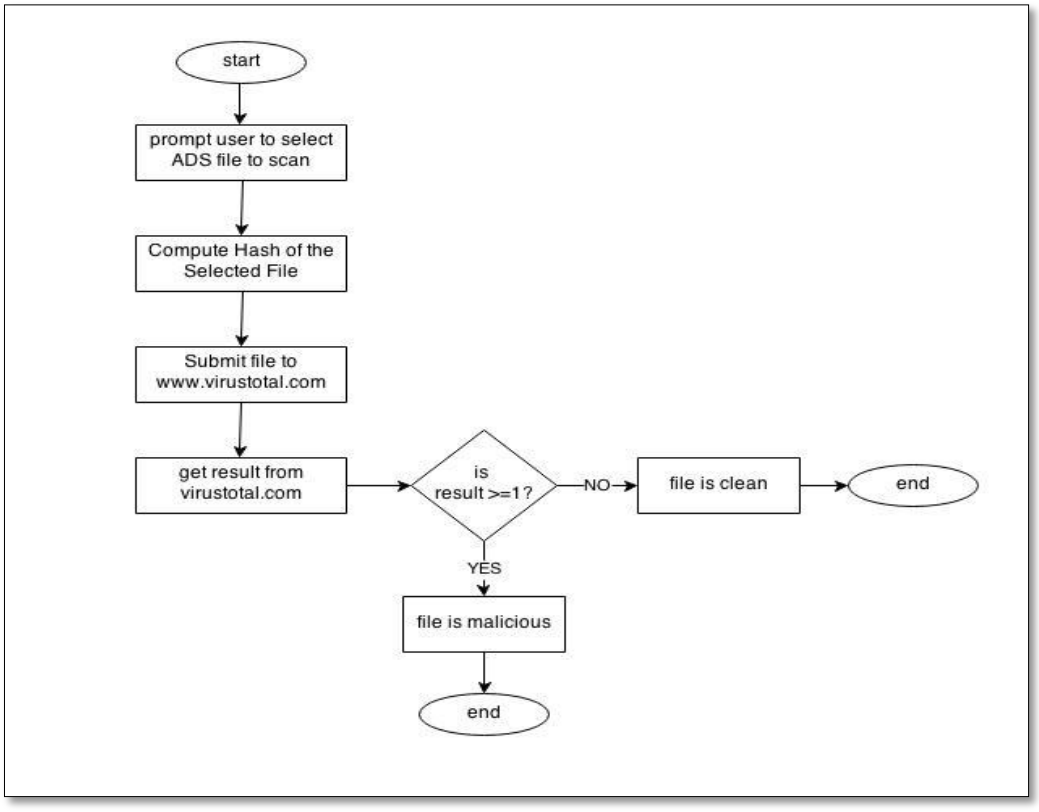

Fig 4.9: Flowchart showing control flow of virus scanner module.

Logic Used: The commonly used antivirus software like Norton, K7Antivirus does not scan Alternate data streams. So scanning ADS for the presence of malicious files is quite difficult. The only way is to scan these files on popular online virus scanning websites containing variety of antiviruses like www.virustotal.com. As it is not possible to transfer ADS over TCP/IP channel so checking an ADS on virustotal.com for the presence of malicious data is not possible. This problem is solved by SADS virus scanner which works by computing the hash value of the selected alternate data stream file and submitting hash value to virustotal.com. Hash value of a file depends on the contents contained in the file. Hash is computed by using md5 as hashing algorithm. This site includes about 52 antiviruses and hash is scanned against them to know whether the file is malicious or not. So there is a rare chance of any malware or virus to go undetected by this scanner.

4.2 Python

The Stealth ADS is implemented using python 2.7. Python is exceptionally a powerful programming language designed by Guido Van Rossum. Python is a scripting language that is very easy to use and install. Now-a-days it is used in various domains for developing applications. It is a dynamic programming language.

The best thing about python is its modularity. There are modules available for almost all the task and this make the python more powerful. Python is an ideal language for hackers who want to code quickly. Comparing python with other languages like C, C++ and java, same task that is done in 1000 lines of codes can be done in few lines.

4.2.1 Features of python: Some of the characteristic features of Python are [41]:

- Simplicity is the main feature of the python language. Code in python appears like some English text and thus naive coder don't need to worry about the technicality of this language. This language has simple and basic Constructs which are very easy to learn.
- Python is a free and open source language. This means python is free to download and install. One can modify the source code of this language according to its need and can define new libraries and modules as per his need.
- While coding in python, one don't need to worry about the internal working of the language like memory consumed by the program etc.

- Python has vast number of libraries available that let one to perform almost every task. Using these Libraries one can performs task like opening a web browser, code a virus, create shell code and much more.
- It has very simple and easy syntax. Python can be easily picked by any person whether he/she is new to programming or is an experienced programmer.
- It is very easy to learn and use as large documentation of python is available. It is open source so freely available.
- It supports both procedural and object oriented programming language.
- Python programs are portable that is they can run in any platform whether it is Linux, windows without making any changes to them.
- Python package includes around thousands of modules that can be utilized easily for developing GUI based applications, website development, games development etc.
- Python provides method for handling errors using exception based error handling.
- Python can be used to run a code very fast that is a code writing in C language can be combined with Python program, and can be run extremely fast.
- Python includes various dynamic data types like Lists and Dictionaries. These are known as high level data types.
- Python code can be easily embedded within C or C++ programs or applications so as to provide a scripting interface.

4.2.2 Python Modules used

The various Python modules that are used in the implementation are explained below in tabular form.

Table 4.1: Description of various Python Modules.

Module name	Description
Tkinter	This is a graphical module for giving graphical look to our python programs. This module helps in building Frames, labels, radio-button and much more.
TtkMessageBox	This module is used for showing Popup or message box in our GUI programs.

OS	This module as name suggests is used for executing operating system commands. It has various useful functions like chdir () - for changing the current directory, getcwd () - for getting the current working directory path, uname ()-gives system version information.
Subprocess	This module allows users to create new process, execute system commands and to obtain their error status code to check whether the command has been executed successfully or not. If subprocess gives status code 0, then it means command has been executed properly with no errors. If status code is 1, then there is an error.
Random	As name suggests this module have a random function that generates random values which can be anything a string or an integer value. This module generates value within a specific range.
TkFileDialog	This module is for adding a file browsing dialog box in our python GUI. This module is useful whenever we want to browse or select a file for opening, saving from the windows directory.
TkFont	This module is for changing the fonts of the text and labels in python graphical programs.
Mechanize	Mechanize is basically used for emulating a web Browser. This module of python is used for connecting to the Internet and performing the entire task in python console as we do in any web browser. In SADS, this module is used for connecting to www.virustotal.com and checking the ads for maliciousness.
Hashlib	This module provide single interface for various powerful secure hash and encoding algorithms. This module is used in proposed system for calculating

	hash function of a file.
bs4 (BeautifulSoup)	This is a HTML Parsing module that is used for manipulating and parsing the contents of any HTML Page. With this module we can retrieve specific information from a Web page like title of a page etc. In the proposed system it is used for extracting the required result after scanning.

CHAPTER 5

EXPERIMENTAL RESULTS

5.1 Improving Stealth of Alternate Data Streams using Encoder

The main purpose of adding encoder in SADS is to increase the stealth of Alternate data streams. By adding encoder not only the data present in Alternate data streams become unreadable but also if any malicious file is hidden in ADS, then it can easily bypass antivirus. Previously it was found out that by compressing the file containing attached ADS, stealth increases [32] and some antivirus can be easily bypassed but using encoder in SADS it has been found out that even more stealth can be increased by hiding data in encoded form. This has been experimentally proved below by using following data files:

File 1: Original netcat file.

File 2: Compressed netcat file

File 3: A compressed file having netcat in ADS

File 4: A compressed file having encoded netcat in ADS

After uploading these four files to www.virustotal.com website, the following results were obtained:

Table 5.1: Comparative results by scanning files using Virustotal site.

Antivirus	**File 1 (27/51)**	**File 2 (23/51)**	**File 3 (8/51)**	**File 4 (0/51)**
AVG	**Detected**	**Detected**	**Detected**	Undetected
AhnLab-V3	**Detected**	Undetected	Undetected	Undetected
AntiVir	**Detected**	**Detected**	Undetected	Undetected
Antiy-AVL	**Detected**	**Detected**	Undetected	Undetected
Bkav	**Detected**	**Detected**	Undetected	Undetected
CMC	**Detected**	**Detected**	Undetected	Undetected
Comodo	**Detected**	**Detected**	**Detected**	Undetected
DrWeb	**Detected**	**Detected**	Undetected	Undetected
ESET-NOD32	**Detected**	**Detected**	Undetected	Undetected
F-Prot	**Detected**	**Detected**	Undetected	Undetected
F-Secure	**Detected**	**Detected**	**Detected**	Undetected

Fortinet	**Detected**	**Detected**	Undetected	Undetected
Jiangmin	**Detected**	**Detected**	Undetected	Undetected
Kaspersky	**Detected**	**Detected**	**Detected**	Undetected
Malwarebytes	**Detected**	**Detected**	Undetected	Undetected
McAfee	**Detected**	Undetected	Undetected	Undetected
McAfee-GW	**Detected**	Undetected	Undetected	Undetected
NANO	**Detected**	**Detected**	Undetected	Undetected
Panda	**Detected**	**Detected**	Undetected	Undetected
Rising	**Detected**	**Detected**	**Detected**	Undetected
Sophos	**Detected**	**Detected**	**Detected**	Undetected
Symantec	**Detected**	**Detected**	Undetected	Undetected
The Hacker	**Detected**	**Detected**	Undetected	Undetected
TrendMicro	**Detected**	**Detected**	**Detected**	Undetected
TrendMicro-HouseCall	**Detected**	**Detected**	**Detected**	Undetected
VIPRE	**Detected**	**Detected**	Undetected	Undetected
ViRobot	**Detected**	Undetected	Undetected	Undetected
Ad-Aware	Undetected	Undetected	Undetected	Undetected
AegisLab	Undetected	Undetected	Undetected	Undetected
Agnitum	Undetected	Undetected	Undetected	Undetected
Avast	Undetected	Undetected	Undetected	Undetected
Baidu-International	Undetected	Undetected	Undetected	Undetected
BitDefender	Undetected	Undetected	Undetected	Undetected
ByteHero	Undetected	Undetected	Undetected	Undetected
CAT-QuickHeal	Undetected	Undetected	Undetected	Undetected
ClamAV	Undetected	Undetected	Undetected	Undetected
Commtouch	Undetected	Undetected	Undetected	Undetected
Emsisoft	Undetected	Undetected	Undetected	Undetected
GData	Undetected	Undetected	Undetected	Undetected
Lkarus	Undetected	Undetected	Undetected	Undetected
K7Antivirus	Undetected	Undetected	Undetected	Undetected
K7Gw	Undetected	Undetected	Undetected	Undetected

Kingsoft	Undetected	Undetected	Undetected	Undetected
MicroWorld-eScan	Undetected	Undetected	Undetected	Undetected
Microsoft	Undetected	Undetected	Undetected	Undetected
Norman	Undetected	Undetected	Undetected	Undetected
Qihoo-360	Undetected	Undetected	Undetected	Undetected
SUPERAntiSpyware	Undetected	Undetected	Undetected	Undetected
TotalDefence	Undetected	Undetected	Undetected	Undetected
VBA32	Undetected	Undetected	Undetected	Undetected
nProtect	Undetected	Undetected	Undetected	Undetected

The above results can be graphically shown as below:

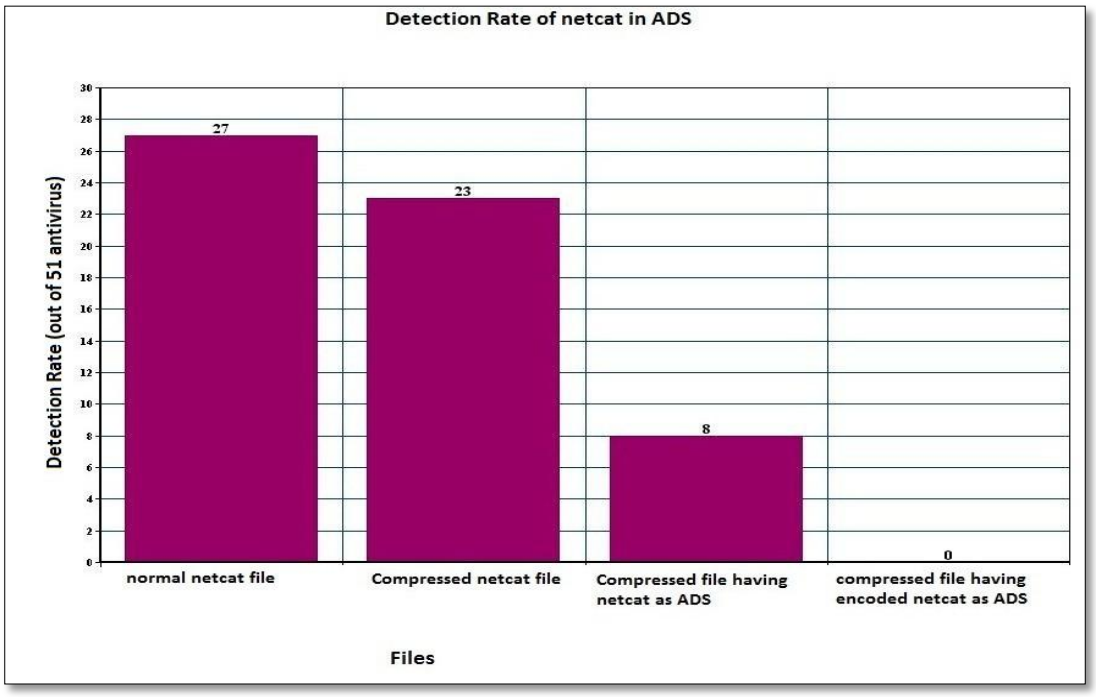

Fig 5.1: Graphical representation of results obtained from virustotal.

From the table and graph it is clearly that encoder added in SADS increases the stealth of Alternate data streams by bypassing all antivirus software effectively. This experiment is also performed on other malicious files also like Passview and eicar_com.exe and same results were generated.

5.2 Comparison of SADS encoder with Existing Encoders

To prove and show how the SADS Encoder is better than existing encoders, three common encoders of metasploit framework are taken. Some comparison tests are performed on the basis of certain parameters like Antivirus Detection Rate, Time Consumed. These parameters are very important for evaluating the encoder's efficiency.

1) Antivirus Detection Rate: This parameter basically means how many antivirus it is able to evade and bypass. Lower the Antivirus Detection rate, more the efficiency of encoder.

2) Encoding Time: Time consumed for encoding a file is another important factor to be considered while selecting an encoder. This parameter is the amount of time spent on encoding a file. Encoding time should be independent of the size of the file.

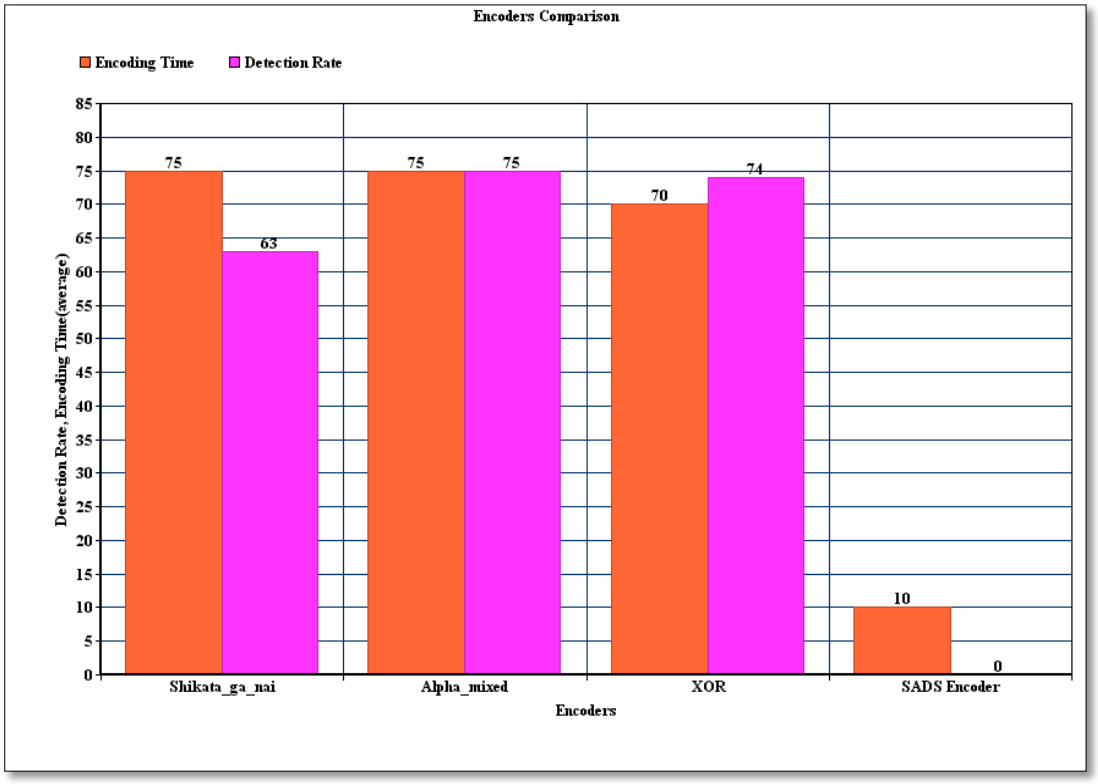

Fig 5.2: Comparing Encoders on the basis of detection rate and encoding time.

From the above graph, it is clear that performance of SADS Encoder is better compared to the other existing encoders.

During comparison test, it has been found that the Encoding time in case of SADS Encoder is constant and is very low. In case of other encoders, the Encoding time increases with the size of the file.

For performing this comparison, 3 malicious files are encoded using different encoders and are analyzed using 52 antiviruses for their detection rate. The Detection Rate is near to Zero in case of SADS Encoder. The test results are shown in table below.

Table 5.2: Comparing detection rate of different encoders for three malicious files.

Encoders	**Malicious Files**		
	Virus.Boot.IsraeliBoot.a	Netcat	Eicar
Shikata_ga_nai	30	33	31
Alpha_mixed	34	36	32
XOR	33	36	33
SADS encoder	0	0	0

5.3 Exploitation of Alternate Data Streams

Hackers can exploit or misuse this strange feature of NTFS by using the Alternate Date Streams for hiding rootkits, backdoors or other hacking tools on an already compromised system without the knowledge of the owner of the system.

Here it is explained how to exploit ADS by hiding a backdoor in any existing file in a system and getting access of that system. Suppose an attacker has once got access of a machine by exploiting any vulnerability. Now when victim's machine will get shut down then attacker will lose the access and attacker needs to perform all the steps for again getting access of victim's machine. So in order to avoid this, attacker will hide a backdoor in victim's machine. The various steps involved are explained below:

1. First of all attacker creates a malicious file named backdoor_win.exe with metasploit for accessing the computer with Windows 7(victim's machine)

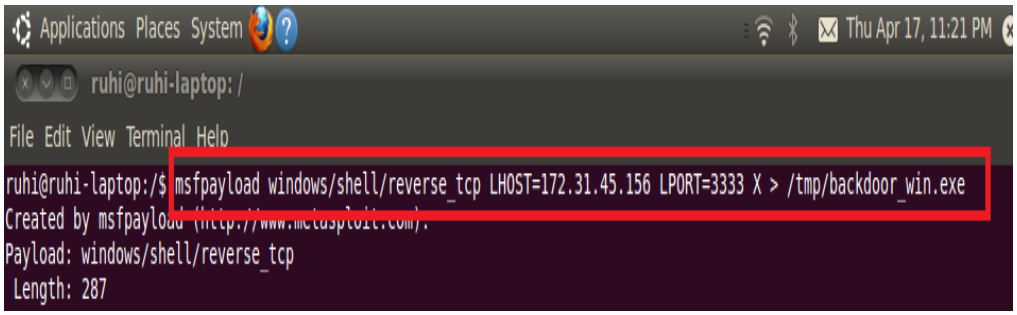

Fig 5.3: Creation of a malicious file using metasploit framework.

2. Now attacker after getting victim's machine access downloads malicious file on victim's machine.

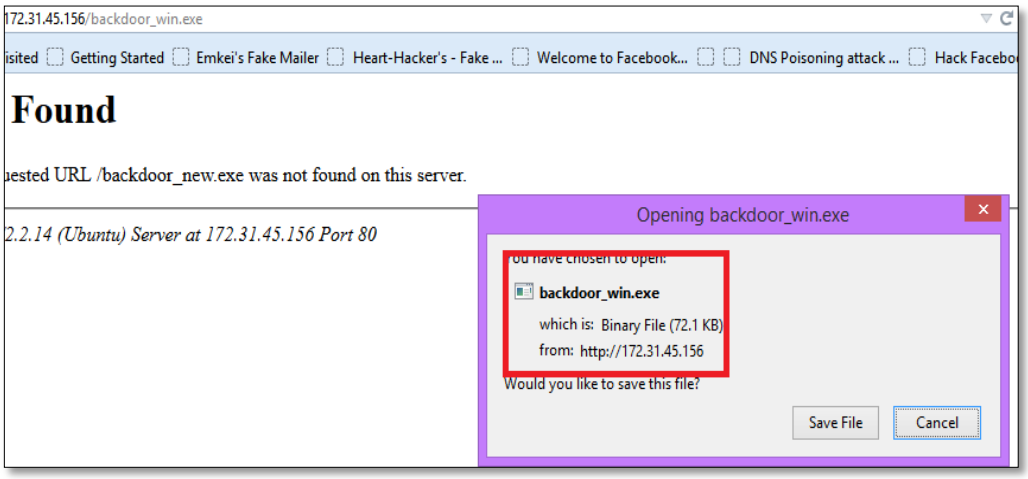

Fig 5.4: Downloading backdoor_win.exe on victim's machine.

3. Now using SADS attacker selects a file and hides the malicious file backdoor_win.exe into it.

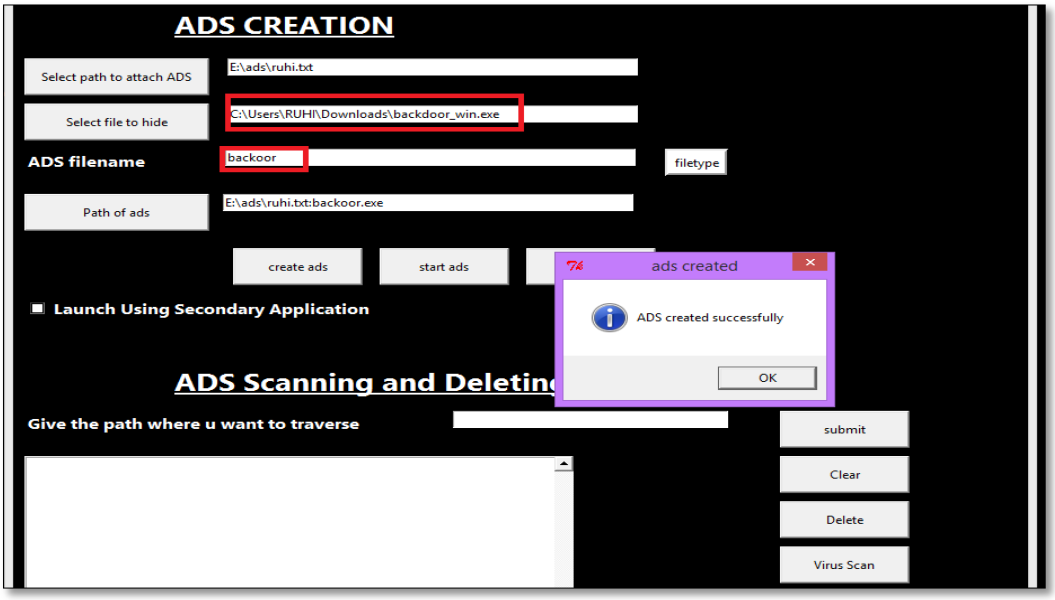

Fig 5.5: Hiding Malicious file in ADS named backdoor.exe.

4. Then attacker starts listener on own system using metasploit framework so that attacker's machine can get shell of victim's system whenever ADS file containing backdoor gets executed on victim's machine.

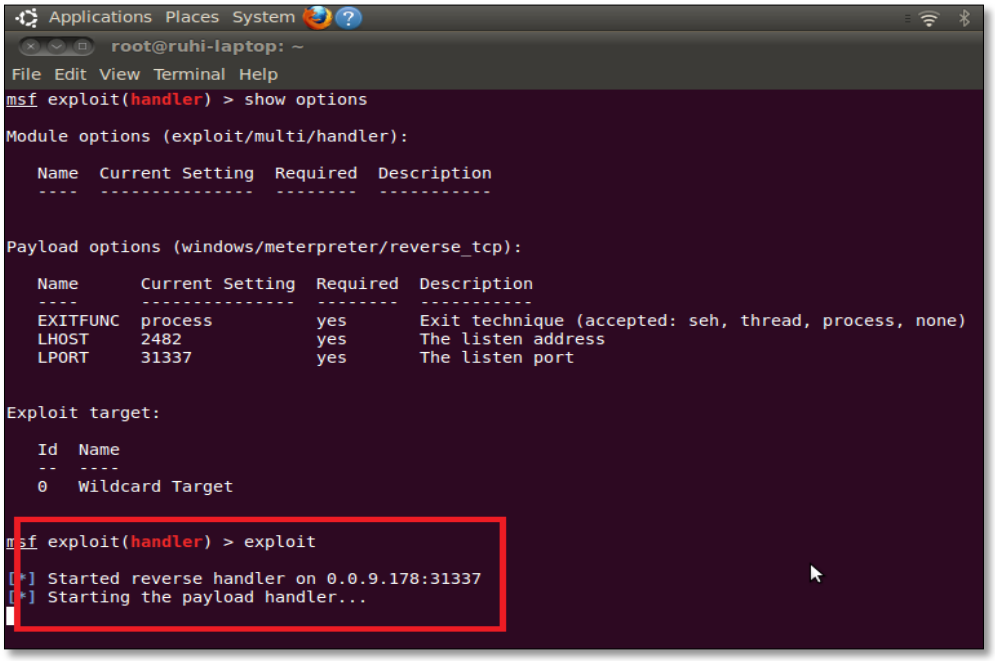

Fig 5.6: Attacker listening to port so as to get shell.

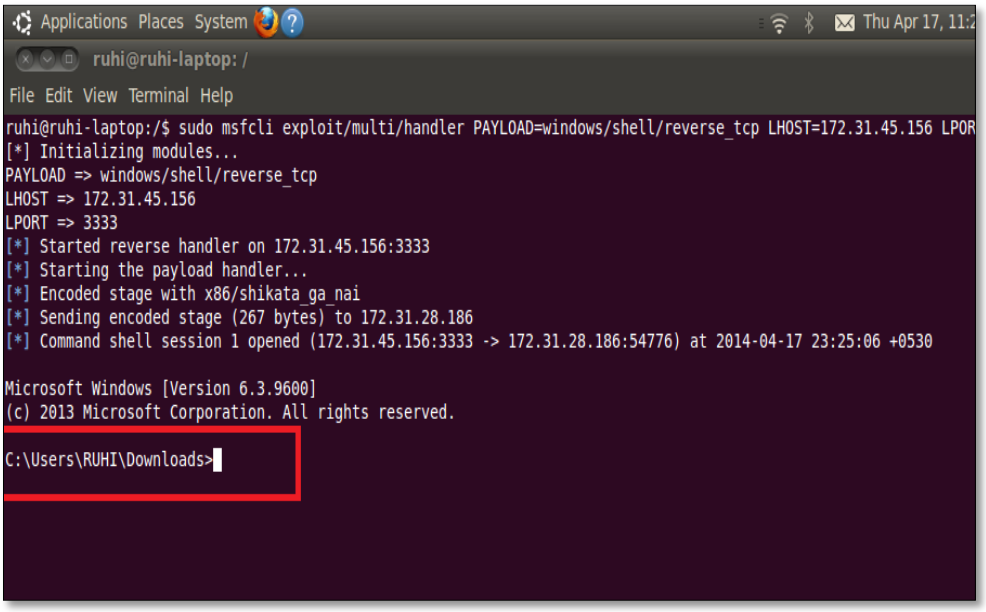

Fig 5.7: Showing attacker has got access of victim's system.

If the attacker wants to get the access of victim's system whenever victim starts machine then attacker needs to write a start-up script on victim machine which can

execute the ADS containing backdoor every time the victim will start machine and if attacker is listening at that time then attacker's machine will get the victim's shell as shown below:

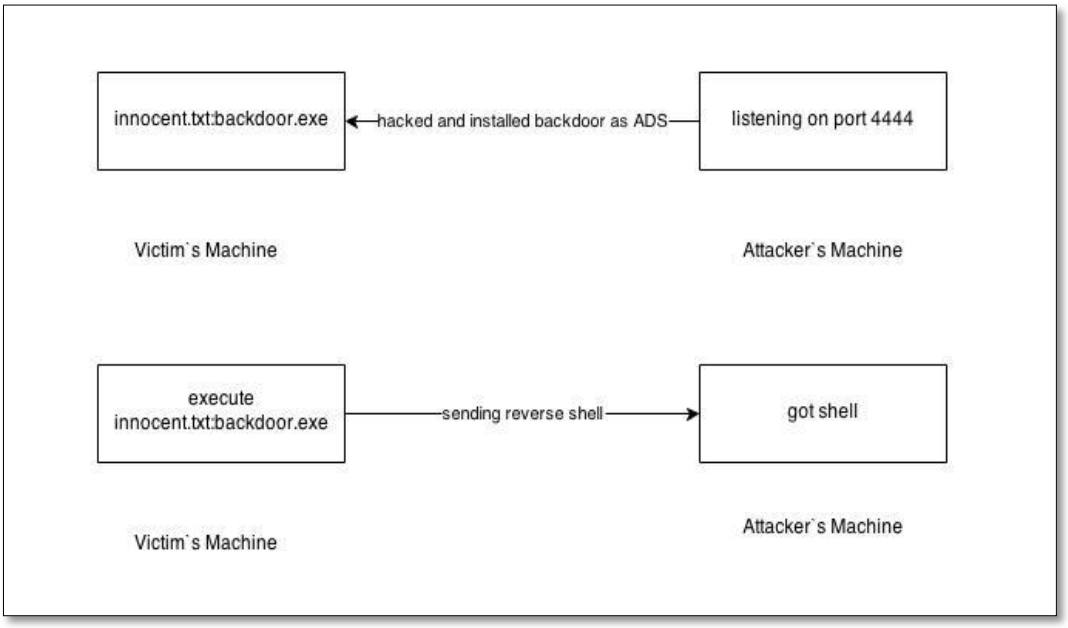

Fig 5.8: Getting shell by executing malicious ADS file.

5.4 Comparison of Stealth ADS with Existing Software

There are few software available in market for ADS detection and deletion. As explained in problem statement no software is available for creation of ADS and for scanning ADS for the presence of malicious file. The comparison table below shows how Stealth ADS is better than other available software:

Table 5.3: Comparison of Stealth Alternate data stream with available software.

Software Name	**ADS Creation**	**ADS Scanning Capability**	**Graphical User Interface**	**ADS deletion Capability**	**Encoder For ADS stealth**	**ADS Virus Scanning**
SADS	Yes	Yes	Yes	Yes	Yes	Yes
Lads	No	Yes	No	No	No	No
Alternate Stream View	No	Yes	Yes	Yes	No	No
LNS	No	Yes	No	No	No	No

Streams	No	Yes	No	Yes	No	No
SFind	No	Yes	No	No	No	No
Ads Spy	No	Yes	Yes	Yes	No	No

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

Alternate data streams is that part of New technology file system which is very less documented, that is, there is very less awareness among people about this feature. Although this feature is widely used still most software does not detect its presence. Alternate data streams provide a great environment for hiding large amount of data behind any legitimate file without the fear of getting detected. The amount of data to be hidden does not depend on the size of the main file. The hidden file can be of any size and any type. So using this Stealth ADS, user can take advantage of hiding personal or sensitive data in an encoded form. Doing this user can secure information from unauthorized attempt.

Alternate data streams can be considered as both a feature and vulnerability of NTFS. Although it has now become vulnerability, still this feature cannot be removed from new technology file system as it is an important feature of that. The only possible way to protect from this vulnerability is to be aware of the presence of these steams in the system and having full knowledge that how these can be used for attacking a system. So user should properly scan their system for the presence of alternate data streams and delete them if found malicious by virus scanner.

6.2 Future Scope

As intruders are now-a-days adding Alternate data streams in compressed files so it becomes difficult to detect them as software used for detection purpose do not detect ADS present in compressed files. In future, support for detecting compressed ADS in Alternate data stream tool that is inspecting inside a compressed file for the presence of any ADS will be provided.

Also the encoder included in tool need to be kept updated in order to maintain the stealth. This can be achieved by making this encoder polymorphic. This means using polymorphic version of SADS Encoder which would produce different signatures patterns every time. Right now SADS deals with signature based antivirus, so in future support for dealing with Heuristic Based Antivirus will be provided in this.

Another feature to be added in the future version of SADS is the offline Malware Scanner that will scan the ADS for maliciousness offline without connecting to the

internet. So user can then choose between any of the two available scanning modes according to the need and resources (internet connectivity) available.

REFERENCES

[1] J.B.D. Joshi, W.G. Aref and A. Ghafoor, "Security models for web-based applications," *Communications of the ACM*, vol. 44, pp. 38-44, Feb. 2001.

[2] J.P. Jesan, "Information security," Magazine Ubiquity, vol. 2006, no. 3, January 2006.

[3] P. Engebretson, *The Basics of Hacking and Penetration Testing*, Syngress, 2011.

[4] K. Graves, "Introduction to Ethical Hacking, Ethics, and Legality," in *Certified Ethical Hacker*, Wiley Publishing, 2007, pp. 1-29.

[5] M. Karresand, "Separating Trojan horses, viruses, and worms-a proposed taxonomy of software weapons," *Information Assurance Workshop, IEEE Systems, Man and Cybernetics Society*, pp. 127-134, 2003.

[6] E. W. Felten, D. Balfanz, D. Dean, and D. S. Wallach. "Web spoofing: An internet con game," Department of Computer Science, Princeton University, Technical Rep. 540-96, 1997.

[7] S. Savage, D. Wetherall, A. Karlin. and T. Anderson, "Practical network support for IP Traceback," *SIGCOMM ACM*, pp. 295-306, 2000.

[8] J. Mirkovic, and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication*, vol. 34, pp. 39-53, April 2004.

[9] C. Macnab, *Network Security Assessment*, 2nd ed., O'Reilly Media, Inc., 2008.

[10] J. Fonseca, M. Vieira, and H. Madeira, "Testing and comparing Web vulnerability scanning tools for SQL injection and XSS attacks," *Dependable Computing, 2007 PRDC 2007, 13th Pacific Rim International Symposium on IEEE*, pp. 365-372 , 2007.

[11] J. Davis, J. MacLean and D. Dampier, "Methods of Information Hiding and Detection in File Systems," *SADFE 2010, Fifth IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering*, Oakland, CA, pp. 66-69, 2010.

[12] H. Berghel, D. Hoelzer and M. Sthultz, "in Data hiding tactics for windows and unix file systems," in *Advances in Computers, vol*. 74, pp 1-17, 2008.

[13] H. Kayarkar and S. Sanyal, "A survey on various data hiding techniques and their comparative analysis," *arXiv preprint arXiv:1206.1957*, 2012.

[14] A. Kumar, and Km. Pooja, "Steganography-A Data Hiding Technique," in *International Journal of Computer applications, vol.* 9, no. 7, pp. 19-23, 2010.

[15] S.M. Thampi, "Information hiding techniques: A tutorial review," ISTE-STTP on Network Security & Cryptography, LBSCE, 2004.

[16] D. Giampaolo, "What is File system," in *Practical File System Design*, Morgan Kaufmann Publishers, Inc., 1999, pp. 7-31.

[17] E. Huebner, D. Bem and C.K. Wee, "Data hiding in the NTFS file system," *Digital Investigation, Elsevier,* vol. 3, pp. 211-226, 2006.

[18] M.E. Russinovich and D. Solomon, "File system," in *Microsoft Windows Internals*, 4th ed., Microsoft Press, 2005, pp. 689-785.

[19] B. Carrier, *File System Forensic Analysis*, vol. 3, Addison-Wesley, 2005, pp. 199-282.

[20] NTFS attribute types [Online]. Available: http://msdn.microsoft.com/en-us/library/dn410153.aspx.

[21] S.H. Mahant, and B. B. Meshram, "NTFS Deleted Files Recovery: Forensics View," *International Journal of Computer Science and Information Technology & Security*, vol. 2, no. 3, pp. 1-7, June 2012.

[22] D. Ben and E.Z. Huebner. "Alternate Data Streams in Forensic Investigations of File Systems Backups," School of Computing and Mathematics University of Western Sydney, pp. 1-11, 2006.

[23] F. Mirza, "Looking for Digital Evidence in Windows," *Biometrics and Security Technologies International Symposium, IEEE,* Islamabad, pp. 1-7, 2008.

[24] M. Broomfield, "NTFS Alternate Data Streams: focused hacking," in *Network Security, Elsevier*, vol. 2006, pp. 7-9, August 2006.

[25] K. Palmgren. (2006). Alternate Data Streams – What's hiding in Your Windows NTFS. White Paper, Global Knowledge Instructor, CISSP, Security+. Available: http://images.globalknowledge.com/wwwimages/whitepaperpdf/WP_DS_Palmgren1.pdf.

[26] M.T. Raggo and C. Hosmer, "Operating System Data Hiding," in *Data Hiding: Exposing Concealed Data in Multimedia, Operating Systems, Mobile Devices and Network Protocols*, Digital Media, Syngress, 2013, pp. 133-164.

[27] Alternate Data Stream: Threat or Menace [online]. Available: http://www.informit.com/articles/article.aspx?p=413685.

[28] NTFS Stream [Online]. Available:

http://msdn.microsoft.com/en-us/library/dn393272.aspx.

[29] Macintosh Hierarchical Filesystem [Online]. Available: http://www.tldp.org/HOWTO/Filesystems-HOWTO-7.html.

[30] E. Sawicki. NTFS Streams [Online]. Available: http://www.biznix.org/whylinux/windows/ads.html.

[31] H. Berghel, and N. Brajkovska, "Wading into alternate data streams," *ACM,* vol.4, no. 4, April 2004.

[32] A.I. Martini, A. Zaharis and C. Ilioudis, "Detecting and Manipulating Compressed Alternate Data Streams in a Forensics Investigation," *Third International Annual Workshop on Digital Forensics and Incident Analysis, IEEE,* pp. 53-59, 2008.

[33] A. Marcella, and R. S. Greenfield, *Cyber forensics: a field manual for collecting, examining, and preserving evidence of computer crimes,* CRC Press, 2001.

[34] V. Kumar, J. Srivastava, and A. Lazarevic, *Managing cyber threats: Issues, approaches, and challenges,* Springer, 2005.

[35] D. Zenkin and E. Kaspersky. NTFS Alternate Data Streams [online]. Available: http://windowsitpro.com/systems-management/ntfs-alternate-data-streams.

[36] R.L. Means, " Alternate data streams: out of the shadows and into the light," 2003.

[37] Zone Identifier ADS's [Online]. Available: http://www.sandersonforensics.com/ Files/ZoneIdentifier.pdf.

[38] D. Martin (2002). Windows, NTFS and Alternate Data Streams. Global Information Assurance Certification Paper. Available: http://www.giac.org/paper/gsec/715/windows-ntfs-alternate-data-streams/101622.

[39] W2K.Streams [Online]. Available: http://service1.symantec.com/sarc/sarc.nsf/html/W2K.Stream.html.

[40] C. Gupta. Dissecting NTFS Streams [online]. Available: http://www.forensicfocus.com/dissecting-ntfs-hidden-streams.

[41] W.J. Chun, *Core python programming,* First ed., Prentice Hall PTR, 2000.

LIST OF PUBLICATIONS

- R. Mahajan, M. Singh, and S. Miglani, "ADS: Protecting NTFS from Hacking", in proceedings of IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014), Jaipur, India, pp. 51, May 09-11, 2014 (is to be published).