Lecture Computer Forensics

Chapter 6: Analysis of an NTFS File System

Harald Baier, Björn Roos

Hochschule Darmstadt, CASED

WS 2011/2012

General information about NTFS

General information about NTFS

The Master File Table (MFT)

General information about NTFS

The Master File Table (MFT)

Attributes

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

NTFS file operations

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

General information about NTFS

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

Foundations of NTFS (1/3)

1. NTFS = New Technologies File System
2. Standard file system of Microsoft operating systems starting from Windows NT/2000
3. Most common file system on end-user computers
4. The maximum theoretical NTFS volume size is $2^{64} - 1$ clusters
5. According to Microsoft (`technet.microsoft.com`) the maximum implemented NTFS volume size is $2^{32} - 1$ clusters
6. Sample case 4 KiB clusters: Max. volume size is
   $$\left(2^{32} - 1\right) \cdot 4 \cdot 2^{10} \text{ Byte} = \left(16 \cdot 2^{40} - 4 \cdot 2^{10}\right) \text{ Byte} \approx 16 \text{ TiB}$$

Foundations of NTFS (2/3)

1. Design principles for NTFS:
   - Reliability
   - Security
   - Support for large storage media
   - Scalability
2. Significantly more complex than FATx
3. No official published specification for NTFS:
   - Numerous detailed unofficial descriptions (e.g. due to Carrier)
   - More and better support for UNIX / Linux (e.g. Debian package `ntfsprogs`)

Foundations of NTFS (3/3)

1. Central paradigm: Everything is a file.
   - Each byte of an NTFS file system belongs to a file.
   - File system data and meta data are located in files, too.
   - Cluster 0 starts at the beginning of the file system.
   - At the beginning of cluster 0 is the boot sector.
2. Cluster size depends on the volume size:

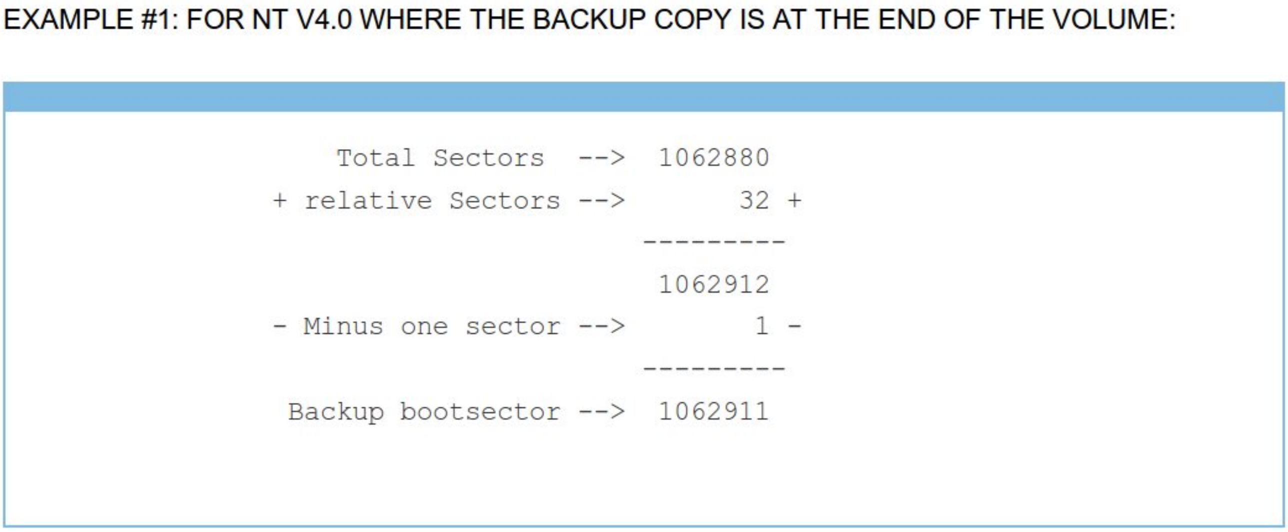
Volume Size	Default Cluster Size
7 megabytes (MB) - 512 MB	512 bytes
513 MB - 1,024 MB	1 KB
1,025 MB - 2 GB	2 KB
2 GB - 2 terabytes	4 KB

Source: `technet.microsoft.com`

Backup Copy of Boot Sector (1/2)

- Useful if boot sector is damaged or overwritten.
- Location?
- Quotes from `support.microsoft.com`:
  - *In NT 3.5x, the second copy is kept in the center of the logical volume (Volume middle).*
  - *In NT 4.0 and Windows 2000, it has been moved to the end of the logical volume (Volume end), ...*
- Brian Carrier, p. 308: *... the sector after the end of the file system, which is where the backup copy is located.*
- Different locations, if partition slack comprises more than one sector, i.e. you should go through this slack space.

Backup Copy of Boot Sector (2/2)



EXAMPLE #1: FOR NT V4.0 WHERE THE BACKUP COPY IS AT THE END OF THE VOLUME:

```
      Total Sectors   -->  1062880
  + relative Sectors -->        32 +
                          ---------
                           1062912
  - Minus one sector -->         1 -
                          ---------
   Backup bootsector -->  1062911
```

Source: `http://support.microsoft.com/kb/153973`

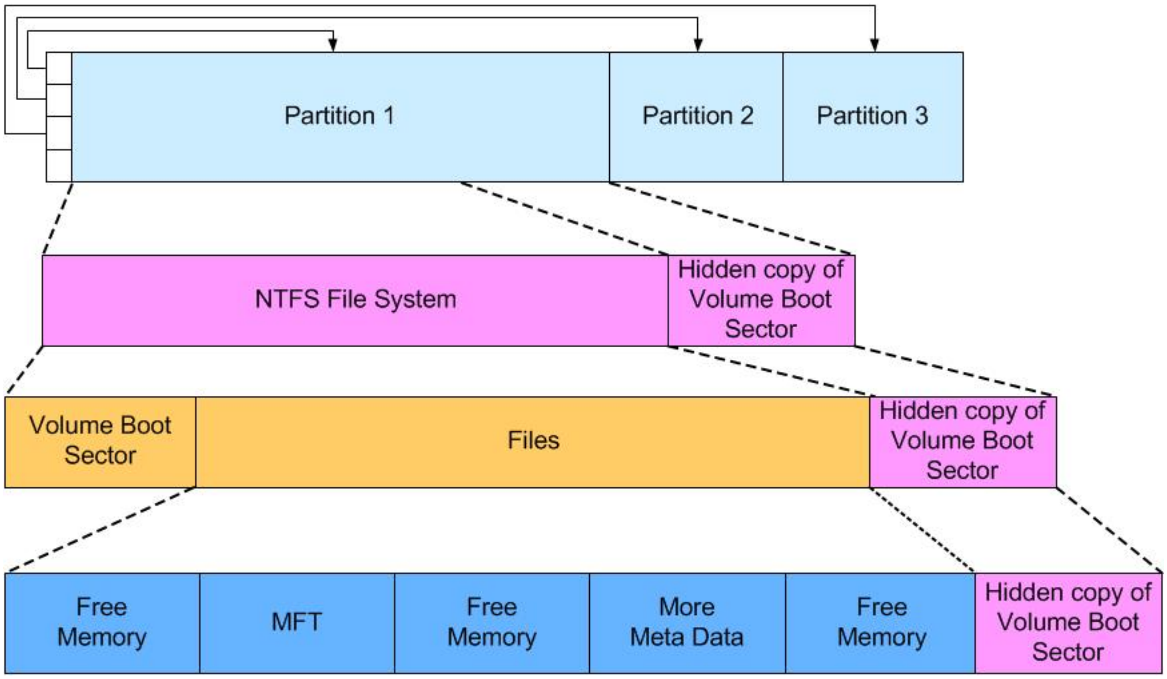
Journaling in NTFS

1. NTFS is a (metadata) journaling file system.
2. File changes are logged → old files retain their validity until writing is marked as successful in the log file.
3. After a system failure restore operations are automatically executed.
4. Journal files:
   - `$LogFile`: NTFS journal file, records metadata changes.
   - `$UsnJrnl`: Change journal, records changes to files.

NTFS Volume Components

Component	Description
NTFS Boot Sector	Stores information about the layout of the volume and the file system structures, as well as the boot code that loads a Windows OS.
Master File Table (MFT)	Contains the information necessary to retrieve files from the NTFS partition, such as the attributes of a file.
File System Data	Stores data that is not contained within the Master File Table.
Master File Table Copy	Includes copies of the records essential for the recovery of the file system if there is a problem with the original MFT.

Overview over the NTFS file system



Adopted from Frank Dotzauer and Reinhard Stampp

General information about NTFS

Time stamps in NTFS

1. Four time stamps are stored:
    - Modified Time: Last write access.
    - Accessed Time: Last read access.
    - Creation Time: File has been created.
    - MFT Modified Time: Last update of MFT record.
2. Time stamps are stored in 64-bit integer values:
    - Number of $0.1 \mu s$ since 1601-01-01, 00:00 UTC.
    - Useful reference date?
    - EPOCH $= 116,444,736,000,000,000$.
3. Caution: *The NTFS file system delays updates to the last access time for a file by up to 1 hour after the last access.* (Source: `msdn.microsoft.com`)

General information about NTFS

File System Meta Data Files

- Store central information about the NTFS file system.
- Their names start with dollar character $ (except for '.').
- At most 16 file system meta data files.
- Sample files:

'Inode' number = Meta data address = MFT record number	File Name
0	`$MFT`
1	`$MFTMirr`
2	`$LogFile`
5	`.`
6	`$Bitmap`
7	`$Boot`

The Master File Table (MFT)

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

The Master File Table (MFT)

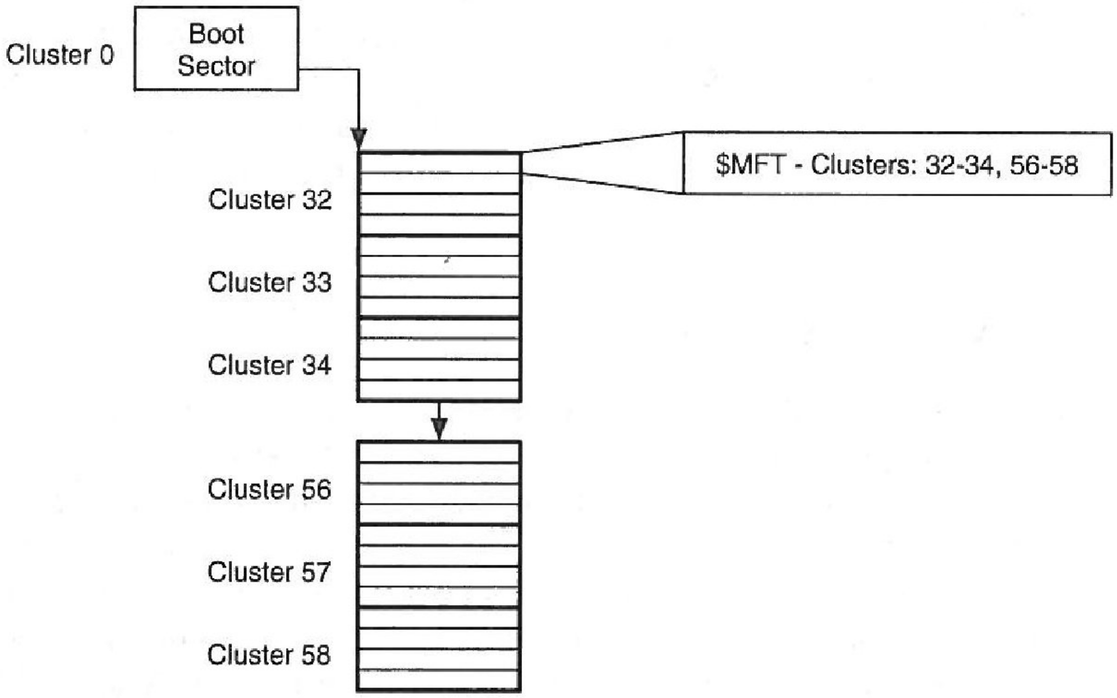
Master File Table (MFT) (1/2)

1. MFT is the heart of NTFS:
   - The MFT is a relational database that consists of rows of file records and columns of file attributes.
   - Every file has at least one entry, including the MFT itself.
   - Small files (up to 700 bytes payload) are stored directly in the MFT.
   - Consists of MFT entries = MFT records.
   - Each MFT record is of same size:
     - Typically 1 KiB.
     - Size of an MFT record is defined in the boot sector.
2. Records are numbered starting from 0.

Master File Table (MFT) (2/2)

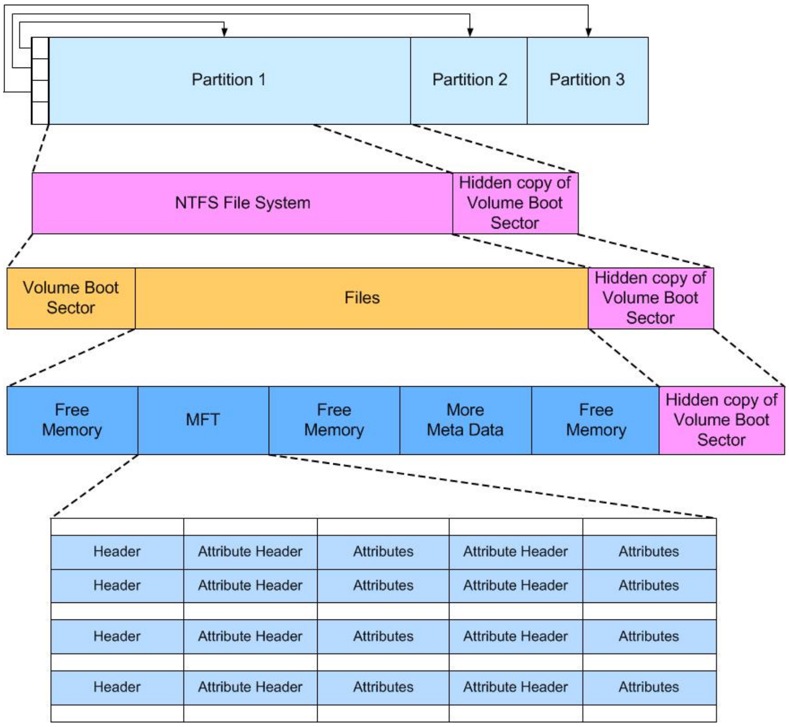
1. Recommendations from Microsoft (`technet.microsoft.com`):
   - At the beginning the MFT is minimal (16 entries for reserved MFT records).
   - Then the MFT grows dynamically if necessary.
   - Therefore it can be fragmented.
2. MFT zone:
   - NTFS reserves typically 12.5% of volume for exclusive use of the MFT.
   - Specific settings may require 25%, 37.5%, or even 50%.
   - Is not used to store user data unless the remainder of the volume becomes allocated.

The boot sector points to the MFT



Source: Brian Carrier, p.275

MFT Record Overview



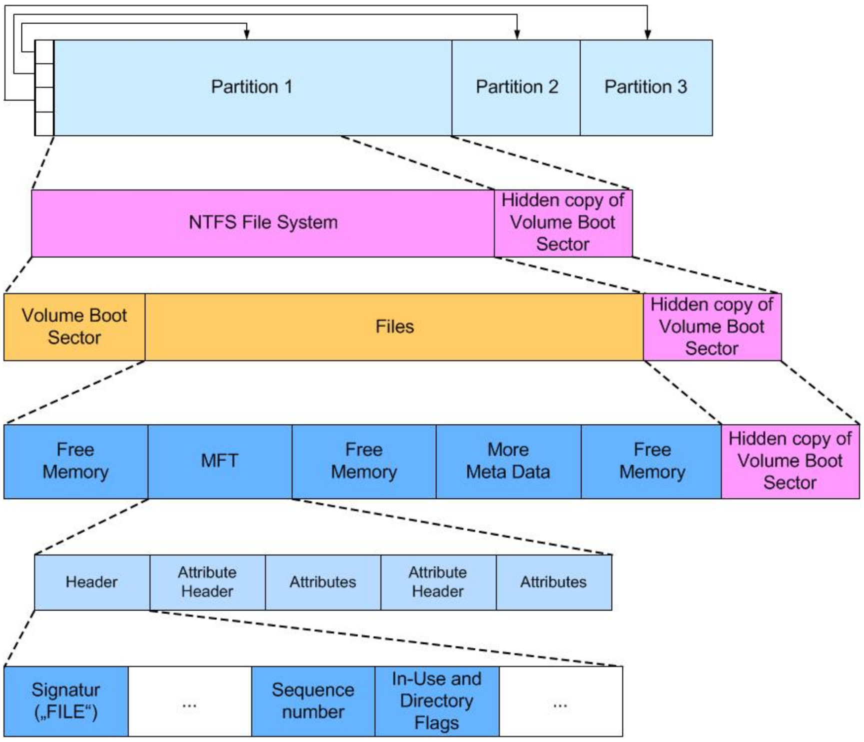
Adopted from Frank Dotzauer and Reinhard Stampp

Areas of an MFT Record

1. MFT entry header:
   - Size: 42 bytes.
   - Structured, 12 fields: 'Signature' (either string 'FILE' or 'BAAD'), allocation status, used bytes in the MFT record, ...
2. MFT entry body:
   - Size: 982 bytes.
   - 'Unstructured', contains MFT attributes.
   - Attributes store e.g. the following:
     - Meta data such as owner, rights, MAC times, ...
     - File name.
     - Content data, encryption keys, ...
   - Unused area.

The Master File Table (MFT)

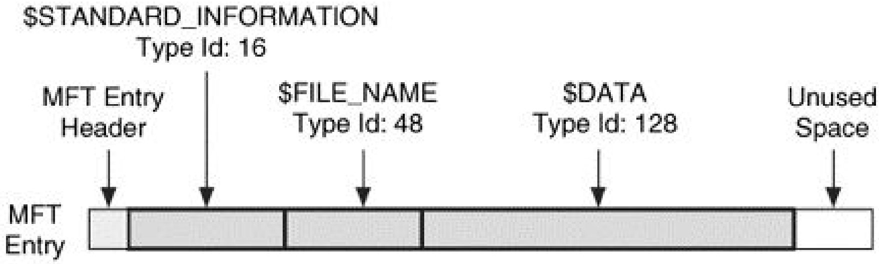
MFT Header Overview



Adopted from Frank Dotzauer and Reinhard Stampp

The Master File Table (MFT)

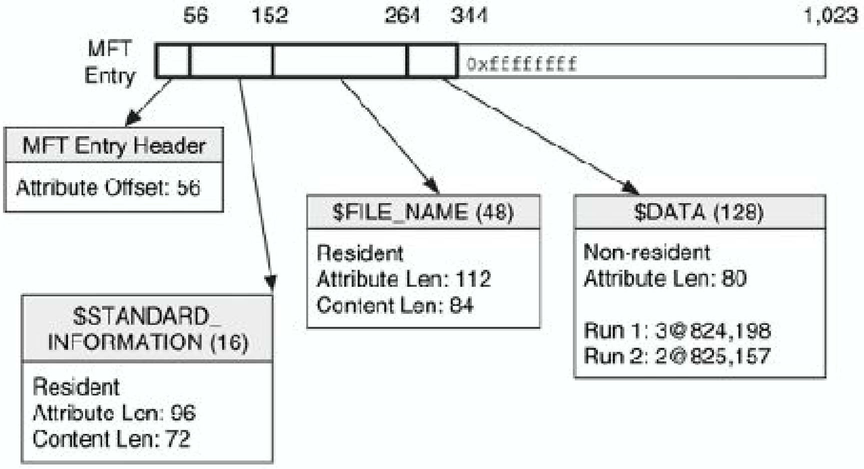
Overview over an MFT entry



Source: Brian Carrier, p.206

The Master File Table (MFT)

A sample MFT entry



Source: Brian Carrier, p.235

The Master File Table (MFT)

Addressing MFT Records

1. Consists of two parts:
   - MFT record number (= 'Inode' number = file number):
     - 48-bit field, i.e. maximum MFT record number: $2^{48} - 1$.
     - Majority not allocated.
     - First entry has an address 0 and is for `$MFT`.
   - Sequence number:
     - 16-bit field
     - Incremented with each new allocation of this record.
2. File reference address := Sequence number || File number
   - 64-bit value.
   - Unique to each generation of a file.
3. Example: 0002 || 0000 0000 → 'Third' generation of `$MFT`.

Attributes

General information about NTFS

The Master File Table (MFT)

Attributes

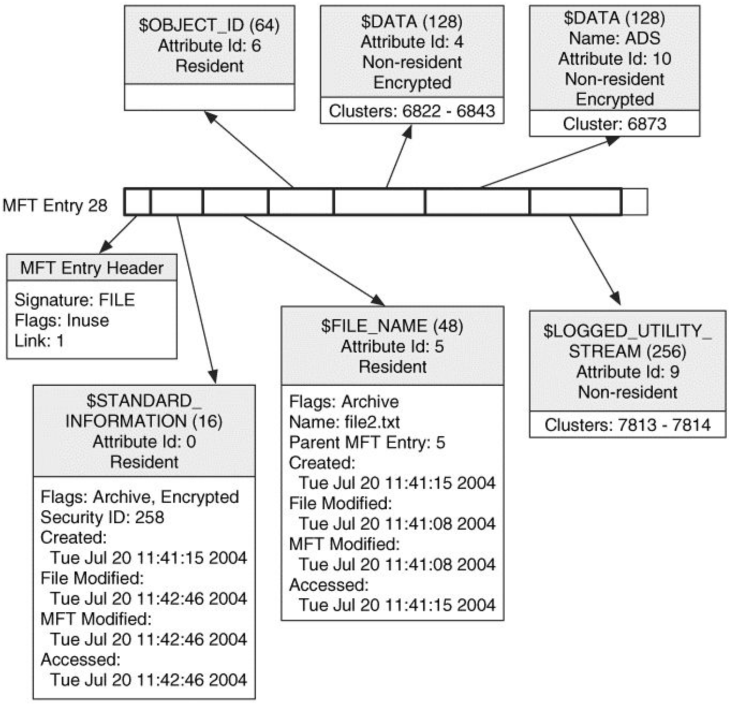
File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

Attributes

Sample MFT Attributes



MFT Entry 28

$OBJECT_ID (64)
Attribute Id: 6
Resident

$DATA (128)
Attribute Id: 4
Non-resident
Encrypted
Clusters: 6822 - 6843

$DATA (128)
Name: ADS
Attribute Id: 10
Non-resident
Encrypted
Cluster: 6873

MFT Entry Header
Signature: FILE
Flags: Inuse
Link: 1

$STANDARD_INFORMATION (16)
Attribute Id: 0
Resident
Flags: Archive, Encrypted
Security ID: 258
Created:
Tue Jul 20 11:41:15 2004
File Modified:
Tue Jul 20 11:42:46 2004
MFT Modified:
Tue Jul 20 11:42:46 2004
Accessed:
Tue Jul 20 11:42:46 2004

$FILE_NAME (48)
Attribute Id: 5
Resident
Flags: Archive
Name: file2.txt
Parent MFT Entry: 5
Created:
Tue Jul 20 11:41:15 2004
File Modified:
Tue Jul 20 11:41:08 2004
MFT Modified:
Tue Jul 20 11:41:08 2004
Accessed:
Tue Jul 20 11:41:15 2004

$LOGGED_UTILITY_STREAM (256)
Attribute Id: 9
Non-resident
Clusters: 7813 - 7814

Source: Brian Carrier, p.230

Two Areas of an MFT Attribute

1. Attribute header: Size is always 16 bytes, fixed structure.
   - Attribute type identifier (e.g. `$DATA`).
   - Size of the attribute.
   - Length of and offset to attribute name (if used).
   - Attribute identifier: To distinguish different attributes of same type identifier (e.g. different `$DATA`-attributes).
2. Attribute body: Size is defined in the header (no fixed structure).
   - Storage of attribute contents.
   - Resident attribute: Attribute payload stored in MFT record.
   - Non-resident attribute: MFT record contains addresses of the data clusters in which attribute content is stored.

Data Structures for an MFT Attribute Header

Key properties:

1. Size is always 16 bytes.
2. Fixed structure.

Address (dec.)	Description	Essential
0 - 3	Attribute type identifier	Yes
4 - 7	Length of attribute	Yes
8 - 8	Non-resident flag	Yes
9 - 9	Length of name	Yes
10 - 11	Offset to name	Yes
12 - 13	Flags	Yes
14 - 15	Attribute identifier	Yes

Attribute Type Identifiers

Attribute name	Value of attribute type identifier (dec.)	4 byte sequence of attribute type identifier
$STANDARD_INFORMATION	16	1000 0000
$ATTRIBUTE_LIST	32	2000 0000
$FILE_NAME	48	3000 0000
$OBJECT_ID	64	4000 0000
$SECURITY_DESCRIPTOR	80	5000 0000
$DATA	128	8000 0000
$LOGGED_UTILITY_STREAM	256	0001 0000

Cluster Runs for Non-Resident Attributes

1. Cluster run: Sequence of consecutive clusters.
2. Aims at storing non-resident attributes.
3. Each cluster run contains the following information:
   - Cluster address of the first cluster of the cluster run.
   - Number of clusters of the cluster run.
4. Example for cluster run of an attribute:
   - Start: 12, Length: 4
   - Start: 80, Length: 3
   - Start: 56, Length: 5

Attributes

Standard Attributes: $STANDARD_INFORMATION

1. Attribute type identifier 16.
2. Always resident.
3. Contains owner, MAC timestamps, flags (hidden, system, ...).
4. Relevant timestamps, as updated at each access.

```
$ istat image-ntfs.dd 0

[REMOVED]
$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 256  ()
Created: Wed Dec 29 20:54:26 2010
File Modified: Wed Dec 29 20:54:26 2010
MFT Modified: Wed Dec 29 20:54:26 2010
Accessed: Wed Dec 29 20:54:26 2010
```

Attributes

Standard Attributes: $FILE_NAME

1. Attribute type identifier 48.
2. Always resident.
3. Contains file name, parent directory, and MAC timestamps.
4. Less relevant timestamp: Time of file creation.

```
$ istat image-ntfs.dd 0

[REMOVED]
$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $MFT
Parent MFT Entry: 5  Sequence: 5
Allocated Size: 16384    Actual Size: 16384
Created: Wed Dec 29 20:54:26 2010
File Modified: Wed Dec 29 20:54:26 2010
MFT Modified: Wed Dec 29 20:54:26 2010
Accessed: Wed Dec 29 20:54:26 2010
```

Standard Attributes: $DATA

1. Attribute type identifier 128.
2. Contains contents of the file.
3. Resident for 'small' files (up to 700 bytes).
4. One standard data field:
   - Has no name (Name displayed in the TSK as `N/A`).
   - File size = Size of standard `$DATA` attribute.
5. Consider further `$DATA` attributes:
   - Alternate Data Streams (ADS, see later).
   - Additional `$DATA` attributes must have names
   - Example: `C:> echo 'Hello world' > out.file:foo` creates ADS to file `out.file` named `foo`.

Standard Attributes: Sample Output

```
$ istat image-ntfs.dd 73

[REMOVED]
Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-3)   Name: N/A   Resident   size: 90
Type: $FILE_NAME (48-2)   Name: N/A   Resident   size: 102
Type: $DATA (128-4)   Name: N/A   Non-Resident, Encrypted   size: 152606
init_size: 152606
29231 29232 29233 29234 29235 29236 29237 29238
29239 29240 29241 29242 29243 29244 29245 29246
29247 29248 29249 29250 29251 29252 29253 29254
29255 29256 29257 29258 29259 29260 29261 29262
29263 29264 29265 29266 29267 29268
Type: $DATA (128-10)   Name: ads2   Non-Resident, Encrypted   size: 23
init_size: 23
16769
Type: $DATA (128-8)   Name: foo   Non-Resident, Encrypted   size: 17
init_size: 17
16764
```

Further Standard Attributes

1. Type identifier 80:
   - $SECURITY_DESCRIPTOR.
   - Set the access rights and other security properties of the file.
2. Type identifier 256:
   - $LOGGED_UTILITY_STREAM.
   - Contains data for encryption/decryption (e.g. EFS): Encrypted File Encryption Key (FEK).
3. Type identifier 176:
   - $BITMAP.
   - Allocation status of data blocks (e.g. for MFT or directory).

Further Attribute Concepts

1. Base MFT entries:
   - Relevant if an MFT entry is too small to store all attributes.
   - First MFT entry is the file's base MFT entry.
   - Base MFT entry contains mapping to attributes in its $ATTRIBUTE_LIST attribute.
2. Sparse Attributes:
   - Cluster that stores only zeros not written to disc.
   - Stored in sparse runs: No start cluster, only length is given.
   - Example:
     - Start: 123, Length: 12
     - Start: —, Length: 5

File System Metadata Files

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

File System Metadata Files

File System Metadata Files (1/3)

1. File system metadata files = meta files:
   - Serve for storing file system data.
   - The first 16 MFT records are reserved for it.
   - Often the user file numbering starts with 'inode' numbers ≥ 16 (e.g. 64).
2. Names of meta files begin with $ and uppercase letters.
3. Records of the MFT:
   - Record 0: `MFT` (record for the MFT itself).
   - Record 1: `$MFTMirr` (backup of the first MFT entries).

File System Metadata Files (2/3)

- Record 2: `$LogFile`
  - Journal for metadata changes.
- Record 3: `$Volume`
  - Contains information on labels, identifier and version of the volume.
- Record 4: `$AttrDef`
  - Values of the type identifier of the attributes used in the file system.
  - Size of the attributes.
  - Chicken-and-egg question: How can we read the `$DATA` attribute of `AttrDef`, if we do not know its type identifier?

File System Metadata Files (3/3)

- Record 5: '.'
  - Root directory of the file system.
- Record 6: `$Bitmap`
  - Allocation status of all clusters in the file system.
- Record 7: `$Boot`
  - Boot sector (contains key information about file system, optionally a boot manager).
  - Always starts in cluster 0.
- Record 9: `$Secure`
  - Definition of security descriptors (SD) of the file system.
  - SDs are used in the `$SECURITY_DESCRIPTOR` attribute of a file to define the file's security status and access controls.

Essential data of the NTFS boot sector ($Boot)

Byte offset (dec.)	Description
11 - 12	Size of an HDD block in bytes
13 - 13	Size of a cluster in HDD blocks
40 - 47	Size of the file system in HDD blocks
48 - 55	Cluster address of the first cluster of the MFT
64 - 64	Size of an MFT record Calculation depends on the sign
68 - 68	Size of an index record in clusters

Example: Boot sector of an NTFS-Partition

```
$ xxd /dev/sdb1 | less

0000000: eb52 904e 5446 5320 2020 2000 0208 0000 .R.NTFS    .....
0000010: 0000 0000 00f8 0000 3f00 ff00 8000 0000 ........?.......
0000020: 0000 0000 8000 0000 ff1f 0300 0000 0000 ................
0000030: 5521 0000 0000 0000 0200 0000 0000 0000 U!..............
0000040: f600 0000 0100 0000 c60a dcee 3fdc ee24 ............?..$
0000050: 0000 0000 fa33 c08e d0bc 007c fb68 c007 .....3.....|.h..
0000060: 1f1e 6866 00cb 8816 0e00 6681 3e03 004e ..hf......f.>..N
0000070: 5446 5375 15b4 41bb aa55 cd13 720c 81fb TFSu..A..U..r...
0000080: 55aa 7506 f7c1 0100 7503 e9dd 001e 83ec U.u.....u.......
0000090: 1868 1a00 b448 8a16 0e00 8bf4 161f cd13 .h...H..........
00000a0: 9f83 c418 9e58 1f72 e13b 060b 0075 dba3 .....X.r.;...u..
```

Interpretation of byte patterns in the boot sector (1/2)

- Size of an HDD block in bytes (11 - 12, essential):
  - Hexdump: _______________
  - Corresponds to the following hex number: _______________
  - Corresponds to the following decimal number: _______________
- Size of a cluster in HDD blocks (13, essential):
  - Hexdump: _______________
  - Corresponds to the following hex number: _______________
  - Corresponds to the following decimal number: _______________

Interpretation of byte patterns in the boot sector (2/2)

1. Address of the first cluster of the MFT (48 - 55, essential):
   - Hexdump: _______________
   - Corresponds to the following hex number: _______________
   - Corresponds to the following decimal number: _______________
2. Size of an MFT record (64, essential):
   - Interpretation as a `signed int` n (using two's complement):
     - Case $n > 0$: Size is given in clusters.
     - Case $n < 0$: Size is 2^{-n} bytes.
   - Hexdump: _______________
   - Corresponds to signed number: _______________
   - Size of an MFT record in bytes: _______________

Access to hexdump of MFT record for $MFT

First tool: `blkcat` (addresses via?)

```
$ blkcat /dev/sdb1 _______  | xxd  |  less
0000000: 4649 4c45 3000 0300 f622 1000 0000 0000  FILE0...."......
0000010: 0100 0100 3800 0100 a001 0000 0004 0000  ....8...........
0000020: 0000 0000 0000 0000 0600 0000 0000 0000  ................
0000030: 0200 0000 0000 0000 1000 0000 6000 0000  ............‘...
```

Second tool: `icat` (addresses via?)

```
$ icat /dev/sdb1 _______  | xxd  |  less
0000000: 4649 4c45 3000 0300 f622 1000 0000 0000  FILE0...."......
0000010: 0100 0100 3800 0100 a001 0000 0004 0000  ....8...........
0000020: 0000 0000 0000 0000 0600 0000 0000 0000  ................
0000030: 0200 0000 0000 0000 1000 0000 6000 0000  ............‘...
```

Access to information within MFT record of $MFT

```
$ istat /dev/sdb1 ____

[REMOVED]
$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $MFT
[REMOVED]
Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-3)   Name: N/A   Resident   size: 74
Type: $DATA (128-1)   Name: $Data   Non-Resident   size: 262144
8533 8534 8535 8536 8537 8538 8539 8540
[REMOVED]
Type: $BITMAP (176-5)   Name: N/A   Non-Resident   size: 4104
8532 8018
```

Access to information within MFT record of $Boot

```
$ istat /dev/sdb1 ____

[REMOVED]
$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $Boot
[REMOVED]
Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 48
Type: $FILE_NAME (48-2)   Name: N/A   Resident   size: 76
Type: $SECURITY_DESCRIPTOR (80-3)   Name: N/A   Resident   size: 116
Type: $DATA (128-1)   Name: $Data   Non-Resident   size: 8192
0 1
```

General information about NTFS

The Master File Table (MFT)

Attributes

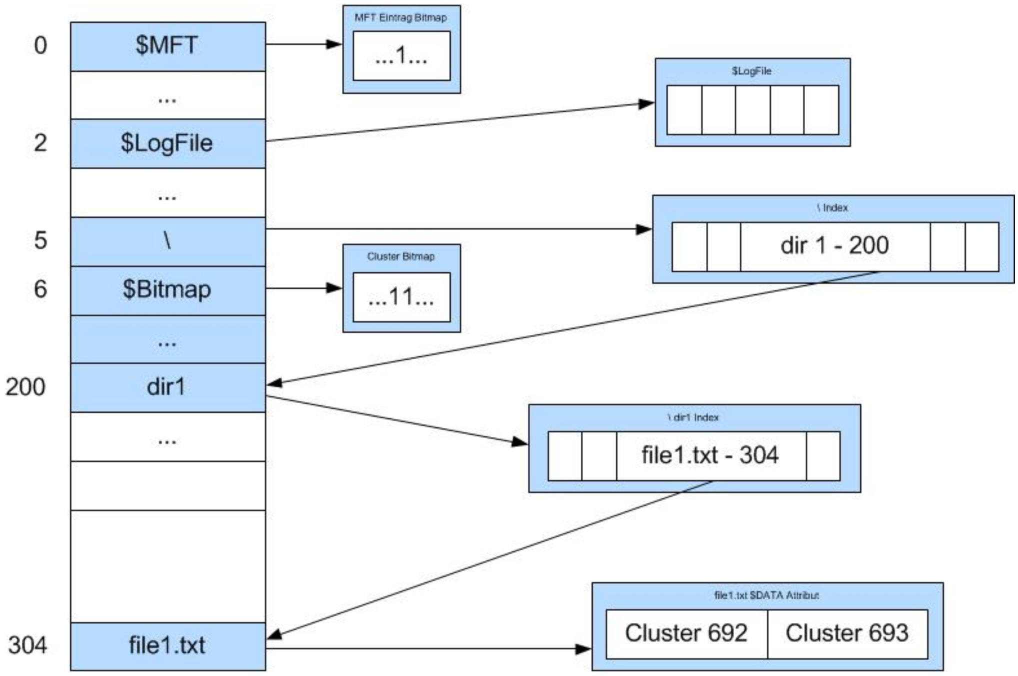
File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

NTFS file operations

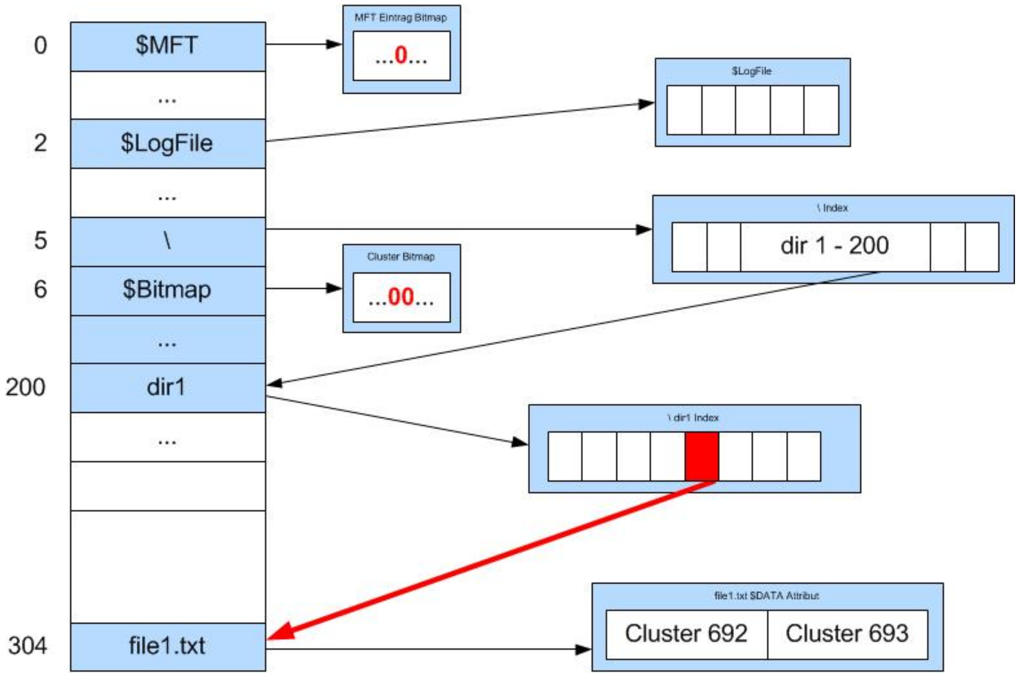
Example: NTFS file creation



Source: Frank Dotzauer and Reinhard Stampp

NTFS file operations

Example: NTFS file deletion



Source: Frank Dotzauer and Reinhard Stampp

Demo: Secure deletion under NTFS

1. Example tool: `shred` (common on Linux)
2. Typical command: `shred -uv file.txt`
3. Sample steps:
   - Create a file.
   - Find out the MFT record number.
   - Note data clusters.
   - Secure deletion with `shred`.
   - Observe the impact on MFT record and data clusters.

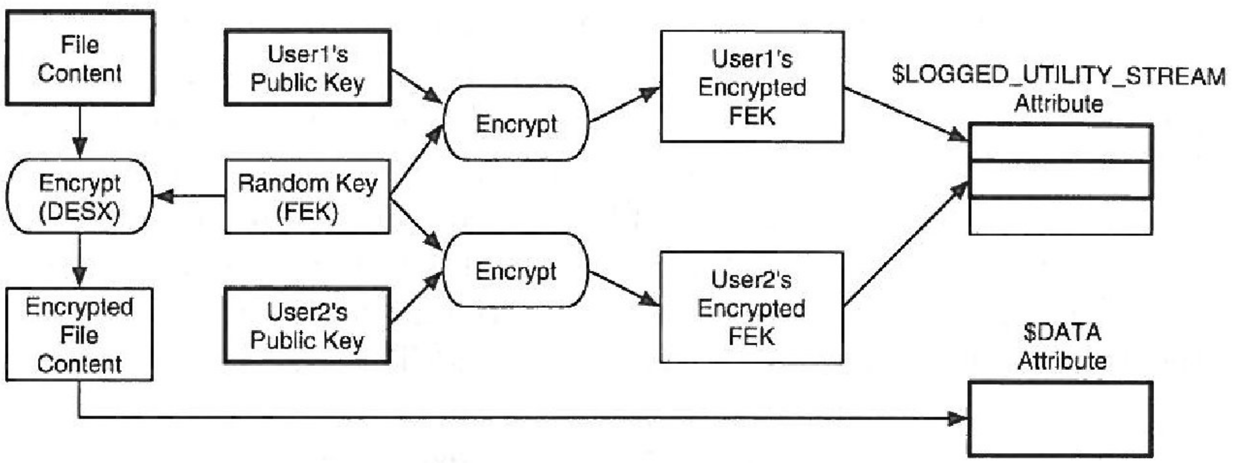
Compression

1. NTFS provides compression on the file system level.
2. Optional units:
   - Single attributes.
   - Files.
   - Folders.
   - Entire volumes.
3. Automatic decompression, if compressed data is read.

Encrypting File System (EFS)

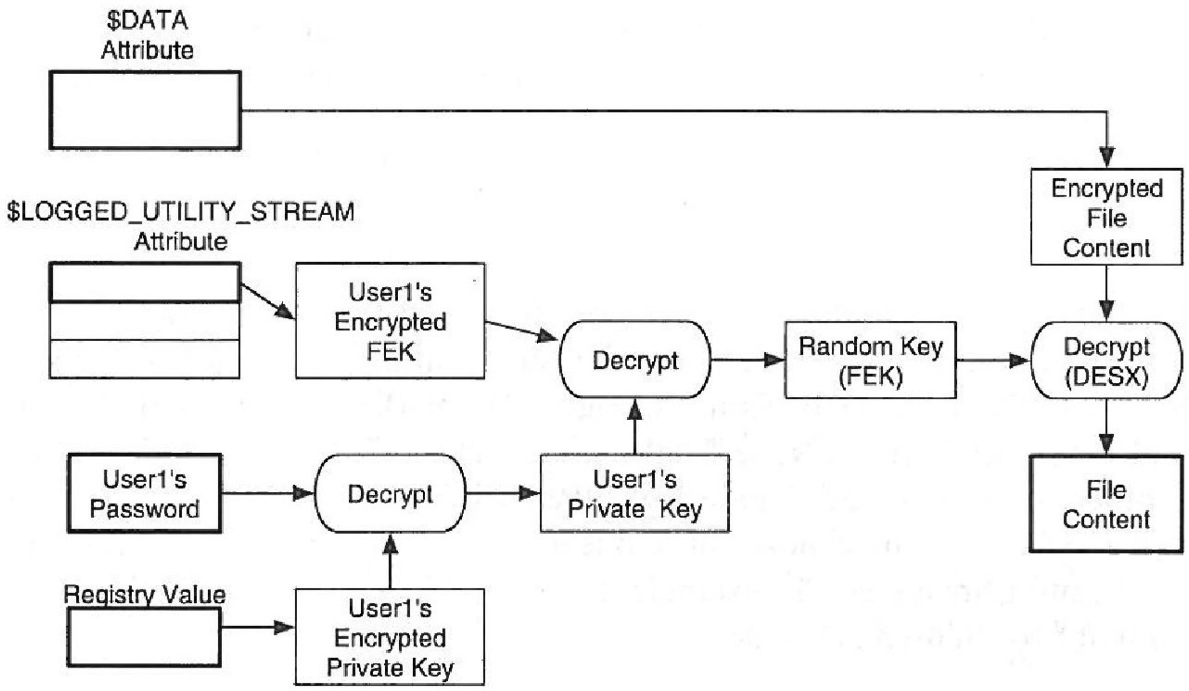
1. Since NTFS v3:
   - Windows 2000.
   - XP Professional (including servers), but not XP Home.
   - Vista, Windows 7.
2. Filesystem level encryption (1024 bit RSA to protect symmetric ephemeral key).
3. Files, directories or drives to encrypt.
4. Elcomsoft claims to recover EFS protected data by using its Advanced EFS Recovery Tool.

EFS Encryption



Source: Brian Carrier, p.210

EFS Decryption



Source: Brian Carrier, p.210

Attributes of EFS protected file

```
$ istat image-suspect.dd 73

[REMOVED]
Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-3)   Name: N/A   Resident   size: 90
Type: $OBJECT_ID (64-11)   Name: N/A   Resident   size: 16
Type: $DATA (128-4)   Name: N/A   Non-Resident, Encrypted   size: 152606
29231 29232 29233 29234 29235 29236 29237 29238
29239 29240 29241 29242 29243 29244 29245 29246
29247 29248 29249 29250 29251 29252 29253 29254
29255 29256 29257 29258 29259 29260 29261 29262
29263 29264 29265 29266 29267 29268
Type: $LOGGED_UTILITY_STREAM (256-6)   Name: $EFS   Non-Resident   size: 688
16767
```

NTFS file operations

Sample Access to File Attributes using TSK

What do the following commands do?

1. `icat partition.dd 0-16`
2. `icat partition.dd 1-48`
3. `icat partition.dd 7-128`
4. `icat partition.dd 7-128-9`
5. `istat partition.dd 111`

Alternate Data Stream (ADS)

General information about NTFS

The Master File Table (MFT)

Attributes

File System Metadata Files

NTFS file operations

Alternate Data Stream (ADS)

Fundamentals to ADS (1/2)

- ADS were introduced into the Windows NTFS file system starting in Windows NT 3.1.
- Each additional stream is stored as an additional `$DATA` attribute.
- A file may have several streams.
- Applications use ADS to store additional information:
  - E.g. under Windows XP SP2 also a zone identifier is stored.
  - Right click on a file → Show its properties (summary page).
- The file system per default only shows one stream, which is the unnamed `$DATA` attribute.

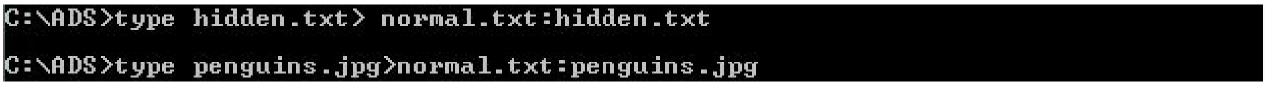
Fundamentals to ADS (2/2)

- All other streams are not visible and not indicated by the Explorer.
- Windows does not consider these streams in the file size.
- Copying a file from an NTFS partition to a FAT partition → All streams (except the main stream) will be lost.
- Sample ADS:
  - A file called *Test.txt* and additional streams *stream1* and *stream2*.
  - Access to stream1: *Text.txt:stream1*
  - Access to stream2: *Text.txt:stream2*

Alternate Data Stream (ADS)

Example

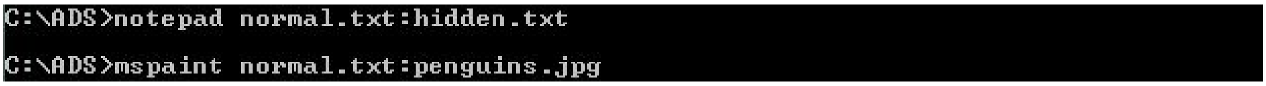
- Create an ADS stream:



```
C:\ADS>type hidden.txt> normal.txt:hidden.txt

C:\ADS>type penguins.jpg>normal.txt:penguins.jpg
```

Hide a text file and a picture

- Show an ADS stream:



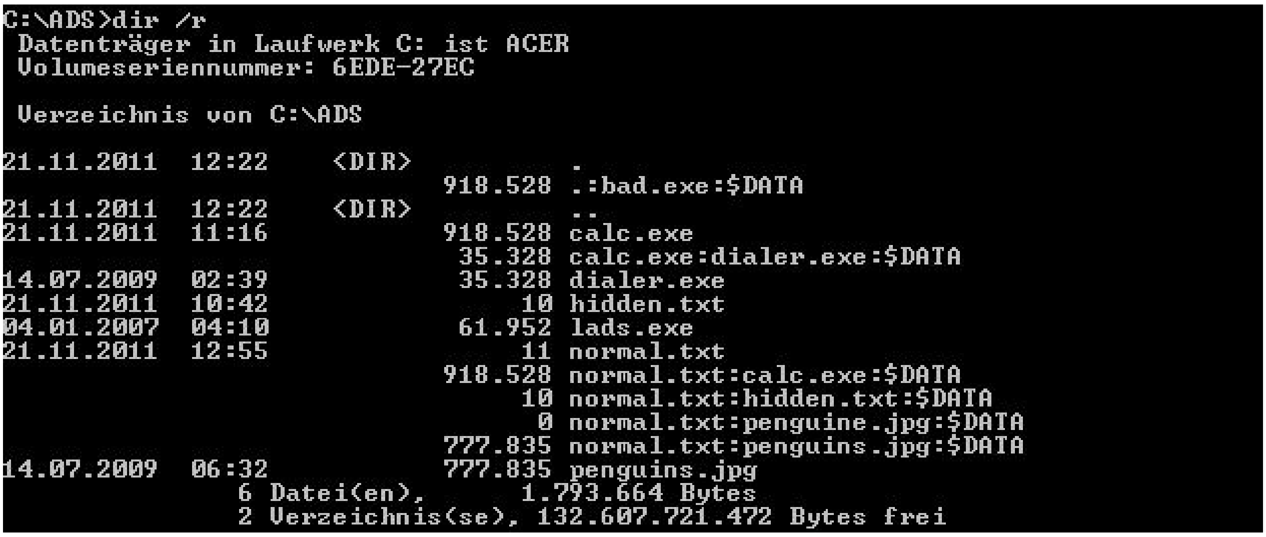
```
C:\ADS>notepad normal.txt:hidden.txt

C:\ADS>mspaint normal.txt:penguins.jpg
```

Show hidden text and picture

Alternate Data Stream (ADS)

Detection of Alternate Data Streams

- Windows 7: dir /r
- Other: Special Tools like `lads`.
- Using TSK (e.g. `fls`).



```
C:\ADS>dir /r
 Datenträger in Laufwerk C: ist ACER
 Volumeseriennummer: 6EDE-27EC

 Verzeichnis von C:\ADS

21.11.2011  12:22    <DIR>          .
                            918.528 .:bad.exe:$DATA
21.11.2011  12:22    <DIR>          ..
21.11.2011  11:16           918.528 calc.exe
                             35.328 calc.exe:dialer.exe:$DATA
14.07.2009  02:39            35.328 dialer.exe
21.11.2011  10:42                10 hidden.txt
04.01.2007  04:10            61.952 lads.exe
21.11.2011  12:55                11 normal.txt
                            918.528 normal.txt:calc.exe:$DATA
                                 10 normal.txt:hidden.txt:$DATA
                                  0 normal.txt:penguine.jpg:$DATA
                            777.835 normal.txt:penguins.jpg:$DATA
14.07.2009  06:32           777.835 penguins.jpg
               6 Datei(en),      1.793.664 Bytes
               2 Verzeichnis(se), 132.607.721.472 Bytes frei
```

ADS and IT-Security

- Data in ADS are exactly like normal executable files. Such data can be executed with a start command.
- Root kits, hacker tools, viruses or files can hide in such streams.
- Some manufacturers of anti-virus software ignore the search in ADS.