**CSCE 4423 – Digital Forensics**

**Lab 2**

## Part A — Read the **VBR** (sector 0 of the NTFS volume)

Extract the geometry and core pointers. All values are **little-endian**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Offset** | **Size** | **Hex** | **Decimal / Interpretation** |
| OEM ID (“NTFS ”) | 0x03 | 8 |  |  |
| **Bytes/sector (BPS)** | **0x0B** | 2 |  |  |
| **Sectors/cluster (SPC)** | **0x0D** | 1 |  |  |
| Reserved sectors | 0x0E | 2 |  |  |
| Total sectors (QWORD) | 0x28 | 8 |  |  |
| **$MFT LCN** | **0x30** | 8 |  |  |
| **$MFTMirr LCN** | **0x38** | 8 |  |  |
| **Clusters per FILE record** | **0x40** | 4 |  | (Signed; if negative: size = 2^ |
| **Clusters per index block** | **0x44** | 4 |  | (Signed; same rule) |
| (If partitioned) StartLBA (from MBR/GPT) | — | — |  |  |

**Compute:**

* BytesPerCluster = BPS × SPC
* LBA($MFT) = StartLBA + LCN($MFT) × SPC

## Part B — Open **$MFT** and jump to the **root directory** (record #5)

1. Go to **LBA($MFT)** from Part A. Confirm records begin with **“FILE”**.
2. Open **record #5** (the **root directory**). It must have **Flags @ +0x16** bit **0x0002** set (directory). It will contain:
   * **0x90 $INDEX\_ROOT** (name **$I30**)
   * possibly **0xA0 $INDEX\_ALLOCATION** (name **$I30**)
   * **0xB0 $BITMAP** (for its index pages)

## Part C — Enumerate the **root folder** via **$I30**

### 4.1 Read **$INDEX\_ROOT (0x90, “$I30”)**

* Inside $INDEX\_ROOT → **INDEX\_HEADER** gives **EntriesOffset**, **IndexEntriesSize**, **Flags**.
* If **Flags & 0x01 (LargeIndex)**, also read **$INDEX\_ALLOCATION**:
  + Decode its runlist; open each **INDX** block (magic “INDX”) and iterate **INDEX\_ENTRY** items there.

### 4.2 Extract each **$I30 index entry**

Each entry begins with an **MFT reference** (8 bytes), lengths, flags, and a **$FILE\_NAME** structure (the key) containing the **UTF-16LE name** and a second set of timestamps.

Fill the table for **EVERY child** listed in root ($I30 entries).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Parent (record #)** | **Child name (from $I30)** | **Child MFT ref** | **Entry flags** | **Notes** |
| 1 | 5 |  |  |  |  |
| 2 | 5 |  |  |  |  |
| 3 | 5 |  |  |  |  |
| 4 | 5 |  |  |  |  |
| 5 | 5 |  |  |  |  |

## Part D — Open each child’s **FILE record** and extract metadata

There are three files and two folders on this drive. For **each item** (the 3 files and 2 folders):

1. Jump to its **FILE record** using the **MFT reference** from $I30 (record # + sequence).
2. List core attributes and values:

**Table — per item metadata**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **MFT Record #** | **In-use?** | **Dir?** | **$SI Created/Modified/MFT-Mod/Access (UTC)** | **$FN Created/Modified/MFT-Mod/Access** | **$DATA resident?** | **Real size (bytes)** | **Runlist (if non-resident)** |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

1. If **file** and **$DATA is non-resident**, decode its **runlist** to find extents:

**Table — data runs ➜ physical placement**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Run #** | **VCN start** | **LCN start** | **Clusters** | **Start LBA (= StartLBA + LCN×SPC)** |
|  | 1 |  |  |  |  |
|  | 2 |  |  |  |  |

Compute **StartLBA** from the partition offset (often 0 on a super-floppy image) and **SPC** from the VBR.

1. If **directory**, list its index placement:

* $INDEX\_ROOT entries present?
* $INDEX\_ALLOCATION present? If yes, decode data runs and list **INDX** block LBAs:
  + LBA(INDX VCN v) = StartLBA + LCN(v) × SPC

## Part E — Deleted file triage (and **bonus** in-place recovery)

One of the 3 files was deleted. Your goals:

### 6.1 **Find** the deleted file

* Scan **$MFT** for an **unused** FILE record (Flags @ +0x16 has **bit 0 cleared**), or search **root’s $I30 slack** (inside INDX pages) for a stale entry that still shows the name & record #.
* Confirm via the file’s **$FILE\_NAME::ParentRef** that it belongs to **root** (record #5) unless it was under one of the folders.

### 6.2 **Assess recoverability**

* If **resident $DATA** → recoverable (bytes are inside the record).
* If **non-resident** → decode the **$DATA runlist** and check volume **$Bitmap** (MFT record #6, attribute 0x80) to see whether those clusters are still **free (0)**:
  + byte = (cluster − 2) // 8, bit = (cluster − 2) % 8.

Document your findings:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Deleted file name (if known)** | **MFT Record #** | **Resident?** | **Data runs (LCN:length)** | **Clusters reused?** | **Verdict** |
|  |  |  |  |  |  |

### 6.3 **Bonus — make it reappear in Explorer**

To undelete **in place**, all three must be true: **MFT record in-use**, **parent $I30 entry live**, **clusters allocated**.

1. **Mark the MFT record in-use**
   * Set **Flags @ +0x16 |= 0x0001** (leave 0x0002=0 for files).
   * In **$MFT::$BITMAP** (record #0, 0xB0), set the **bit for this record #** to **1**.
2. **Re-insert or repair the parent’s $I30 entry**
   * In root’s FILE record (or appropriate parent), add an **INDEX\_ENTRY** for this file (with correct **FileReference** and **$FILE\_NAME** key).
   * If the directory uses **$INDEX\_ALLOCATION**, allocate an **INDX** slot and set the bit in **$I30::$BITMAP (0xB0)**.
3. **Mark the data clusters allocated** (non-resident only)
   * In **$Bitmap** (record #6), set the bits for each cluster in the runlist to **1**.
4. **Remount** and verify the file now appears.

**What to submit for the bonus:** offsets you changed, before/after hex, and a screenshot of the file visible in Explorer.