Windows Kernel Internals

NTFS

David B. Probert, Ph.D.
Windows Kernel Development
Microsoft Corporation
Basic Design Points

- Aries Logging
- Meta-data via Cache Manager
- Self describing meta-data
- B-trees for fast index lookup
- Multiple user data streams
Disk Basics

• Volume exported via device object
• Addressed by byte offset and length
• Enforced on sector boundaries
• NTFS allocation unit - clusters
• Round size down to clusters
NTFS Knows Files

- Partition is collection of files
- Common routines for all meta-data
- Utilizes MM and Cache Manager
- No specific on-disk locations
Some System Files

- $Bitmap
- $BadClus
- $Boot
- . (root directory)
- $Logfile
- $Volume
MFT File

- Data is entirely File Records
- File Records are fixed size
- Every file on volume has a File Record
- File records are recycled
- Reserved area for system files
File Records

• ‘Base’ file record for each file
• Header followed by ‘Attributes’
• Additional file records as needed
• Update Sequence Array
• ID by offset and sequence number
File Basics

• Timestamps
• File attributes (DOS + NTFS)
• Filename (+ hard links)
• Data streams
• ACL
• Indexes
File Building Blocks

• File Records
• Ntfs Attributes
• Allocated clusters
File Record Header

- USA Header
- Sequence Number
- First Attribute Offset
- First Free Byte and Size
- Base File Record
- IN_USE bit
NTFS Attributes

- Type code and optional name
- Resident or non-resident
- Header followed by value
- Sorted within file record
- Common code for operations
<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$STANDARD_INFORMATION</td>
<td>(Time Stamps, DOS Attributes)</td>
</tr>
<tr>
<td>$FILE_NAME - VeryLongFileName.Txt</td>
<td></td>
</tr>
<tr>
<td>$FILE_NAME - VERYLO~1.TXT</td>
<td></td>
</tr>
<tr>
<td>$DATA (Default Data Stream)</td>
<td></td>
</tr>
<tr>
<td>$DATA - “VeryLongFileName.Txt:A named stream”</td>
<td></td>
</tr>
<tr>
<td>$END</td>
<td>(Available for attribute growth or new attribute)</td>
</tr>
</tbody>
</table>
Attribute Header

- Length
- Form
- Name and name length
- Flags (Compressed, Encrypted, Sparse)
Resident Attributes

• Data follows attribute header
• ‘Allocation Size’ on 8-byte boundary
• May grow or shrink
• Convert to non-resident
Non-Resident Attributes

• Data stored in allocated disk clusters
• May describe sub-range of stream
• Sizes and stream properties
• Mapping pairs for on-disk runs
Some Attribute Types

$STANDARD_INFORMATION
$FILE_NAME
$SECURITY_DESCRIPTOR
$DATA
$INDEX_ROOT
$INDEX_ALLOCATION
$BITMAP
$EA
Mapping Pairs

- Stored in a byte optimal format
- Represents allocation and holes
- Each pair is relative to prior run
- Used to represent compression/sparse
Indexes

- File name and view indexes
- Indexes are B-trees
- Entries stored at each level
- Intermediate nodes have down pointers
- $INDEX_ROOT
- $INDEX_ALLOCATION
- $BITMAP
Index Implementation

- Top level - $INDEX_ROOT
- Index buckets - $INDEX_ALLOCATION
- Available buckets - $BITMAP
$INDEX_ROOT

E  J  R  end

A B C  G I  N P Q  Z

$INDEX_ALLOCATION

unused  G I  A B C  data  Z  N P Q

$BITMAP

0x36 (00110110)
$ATTRIBUTE_LIST

• Needed for multi-file record file
• Entry for each attribute in file
• Resident or non-resident form
• Must be in base file record
Attribute List (example)

- **Base Record - 0x200**
  - 0x10 - Standard
  - 0x20 - Attribute List
  - 0x30 - FileName
  - 0x80 - Default Data
  - 0x80 - Data1 “Owner”

- **Aux Record - 0x180**
  - 0x30 - FileName
  - 0x80 - Data “Author”
  - 0x80 - Data0 “Owner”
  - 0x80 - Data “Writer”
<table>
<thead>
<tr>
<th>Code</th>
<th>FR</th>
<th>VCN</th>
<th>Name</th>
<th>(Not Present)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x10</td>
<td>0x200</td>
<td>0</td>
<td>$Standard</td>
<td>$Data</td>
</tr>
<tr>
<td>0x30</td>
<td>0x200</td>
<td>0</td>
<td>$Filename</td>
<td></td>
</tr>
<tr>
<td>0x30</td>
<td>0x180</td>
<td>0</td>
<td>$Filename</td>
<td></td>
</tr>
<tr>
<td>0x80</td>
<td>0x200</td>
<td>0</td>
<td>$Data</td>
<td></td>
</tr>
<tr>
<td>0x80</td>
<td>0x180</td>
<td>0</td>
<td>“Author”</td>
<td>$Data</td>
</tr>
<tr>
<td>0x80</td>
<td>0x180</td>
<td>0</td>
<td>“Owner”</td>
<td>$Data</td>
</tr>
<tr>
<td>0x80</td>
<td>0x200</td>
<td>40</td>
<td>“Owner”</td>
<td>$Data</td>
</tr>
<tr>
<td>0x80</td>
<td>0x180</td>
<td></td>
<td>“Writer”</td>
<td>$Data</td>
</tr>
</tbody>
</table>
Discussion