CS 537
Lecture 17
NTFS internals

Michael Swift

NTFS Recoverability

PC disk I/O in the old days: speed was most important

NTFS changes this view – Reliability counts most:

- I/O operations that alter NTFS structure are implemented as atomic transactions
  - Change directory structure,
  - extend files, allocate space for new files
- Transactions are either completed or rolled back
- NTFS uses redundant storage for vital FS information
  - Contrast with FAT / HPFS on-disk structures, which have single sectors containing critical file system data
  - Read error in these sectors --> volume lost

NTFS On-Disk Structure

- Volumes correspond to logical partitions on disk
- Fault tolerant volumes may span multiple disks
  - Same block stored on separate disks for redundancy
- Volume consists of series of files + unallocated space
  - FAT volume: some areas specially formatted for file system
  - NTFS volume: all data are stored as ordinary files
    - e.g., inodes stored in files
- NTFS refers internally to clusters
  - Cluster factor: #sectors/cluster; varies with volume size (integral number of physical sectors; always a power of 2)
- Logical Cluster Numbers (LCNs):
  - refer to physical location == block number in Unix
  - LCNs are contiguous enumeration of all clusters on a volume

Master File Table

All data stored on a volume is contained in a file

- MFT: Heart of NTFS volume structure
  - Implemented as array of file records
  - One row for each file on the volume (including one row for MFT itself)
  - Metadata files store file system structure information (Hidden files: $MFT; $Volume,...)
  - More than one MFT record for highly fragmented files
  - Nfi.exe Utility from OEM Support Tools allows to dump MFT content (see support.microsoft.com/support/kb/articles/Q253/0/66.asp)

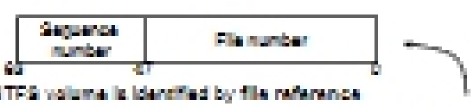
NTFS metadata

- NTFS writes to log file ($LogFile)
  - Record all commands that change volume structure
- Root directory
  - When NTFS tries to open a file, it starts search in the root directory
  - Once the file is found, NTFS stores the file's MFT file reference
  - Subsequent read/write ops. may access file's MFT record directly
- Bitmap file ($Bitmap):
  - stores allocation state of volume; each bit represents one cluster
- Boot file ($Boot):
  - Stores bootstrap code
  - Has to be located at special disk address
  - Represented as file by NTFS → file ops. possible (!) (no editing)

NTFS metadata (contd.)

- Bad-cluster file ($BadClus)
  - Records bad spots on the disk
- Volume file ($Volume)
  - Contains: volume name, NTFS version
  - Bit, which indicates whether volume is corrupted
- Attribute Definition Table ($AttrDef)
  - Defines attribute types supported on the volume
  - Indicates whether they can be indexed, recovered, etc.

File Records & File Reference Numbers



- File on NTFS volume is identified by file reference
  - File number == index in MFT
  - Sequence number – used by NTFS for consistency checking; incremented each time a reference is re-used
  - Like inode numbers, but need not be allocated; just assigned
- File Records: File is collection of attribute/value pairs
  - Unnamed data attribute
  - Other attributes: filename, time stamp, security descriptor,...
  - Each file attribute is stored as separate stream of bytes within a file

File Records (contd.)

- NTFS doesn't read/write files:
  - It reads/writes attribute streams
  - Operations: create, delete, read (byte range), write (byte range)
  - Read/write normally operate on unnamed data attribute
- Record in MFT contains set of attributes
  - attributes are named
  - can be stored in MFT or externally

Standard Attributes for NTFS Files

Attribute	Description
Standard information	File attributes: read-only, archive, etc; time stamps; creation/modification time; hard link count
Filename	Name in Unicode characters; multiple filename attributes possible; short names for access by MS-DOS and 16-bit Win applications
Security descriptor	Specifies who owns the file and who can access it
data	Contents of the file; a file has one default unnamed data attribute; directory has no default data attrib.
Index root, Index	Three attributes used to implement filename allocation, btree index for large directories (dirs. only)
Attribute list	List of attributes that make up the file and file reference of the MFT record in which the attribute is located (for files which require multiple MFT file records)

Attributes (contd.)

- Each attribute in a file record has a name and a value
- NTFS identifies attributes:
  - Uppercase name starting with $: $FILE_NAME, $DATA
- Attribute's value: Byte stream
  - The filename for $FILE_NAME
  - The data bytes for $DATA
- Attribute names correspond to numeric typecodes
- File attributes in an MFT record are ordered by typecodes
  - Some attribute types may appear more than once (e.g. Filename)

Filenames

- Name parsing, including wildcards, is handled by NTFS
  - In Linux, shell expands wildcards
- Directory traversal is handled by NTFS
  - The kernel hands it a full pathname
  - In Linux, an FS only expands one name at a time

Resident & Nonresident Attributes

- Small files:
  - All attributes and values fit into MFT
  - Attribute with value in MFT is called "resident"
  - All attributes start with header (always resident)
  - Header contains offset to attr. value and length of value
  - Data is contained within data attribute
    - Very efficient for small files – no extra seeks/reads