OPERATING SYSTEMS

Input - Output Management and Security - 10

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

OPERATING SYSTEMS

Case Study : Windows File System - 2

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

OPERATING SYSTEMS

Course Syllabus - Unit 5

10 Hours

Unit-5:Unit 5: IO Management and Security

I/O Hardware, polling and interrupts, DMA, Kernel I/O Subsystem and Transforming I/O Requests to Hardware Operations - Device interaction, device driver, buffering System Protection: Goals, Principles and Domain of Protection, Access Matrix, Access control, Access rights. System Security: The Security Problem,Program Threats, System Threats and Network Threats. Case Study: Windows 7/Windows 10

OPERATING SYSTEMS

Course Outline

47	I/O Hardware, polling and interrupts	13.1,13.2
48	DMA	13.2.3
49	Transforming I/O Requests to Hardware Operations, Device interaction, device driver, buffering.	13.5
50	Goals, Principles and Domain of Protection	14.1-14.3
51	Access Matrix	14.4
52	Access control, Access rights	14.5-14.7
53	The Security Problem	15.1
54	Program Threats	15.2
55	System Threats and Network Threats	15.3
56	Case Study : Windows File System	17.5

- # Case study : Windows File System - 2

Case study : Windows File System - Recovery

- In many simple file systems, a power failure at the wrong time can damage the file-system data structures so severely that the entire volume is scrambled.

- Many UNIX file systems, including UFS but not ZFS , store redundant metadata on the disk, and they recover from crashes by using the fsck program to check all the file-system data structures and restore them **forcibly** to a **consistent state**.

- Restoring them often involves deleting damaged files and freeing data clusters that had been written with user data but not properly recorded in the file system's metadata structures.

Case study : Windows File System - Recovery

- NTFS takes a different approach to file-system robustness.
- In NTFS , all file system data-structure updates are performed inside transactions.
- Before a data structure is altered, the transaction writes a log record that contains redo and undo information.
- After the data structure has been changed, the transaction writes a commit record to the log to signify that the transaction succeeded
- After a crash, the system can restore the file-system data structures to a consistent state by processing the log records, first redoing the operations for committed transactions and then undoing the operations for transactions

Case study : Windows File System - Recovery

- Periodically usually every 5 seconds, a checkpoint record is written to the log.
- The first time after system startup that an NTFS volume is accessed, NTFS automatically performs file-system recovery.
- Does not guarantee that all the user-file contents are correct after a crash.
- It ensures only that the file-system data structures (the metadata files) are undamaged and reflect some consistent state that existed prior to the crash.
- Current Versions of Windows does ensure the user files as well

Case study : Windows File System - Recovery

- The log has two sections
  - **Logging Area** which is a circular queue of log records,
  - **Restart Area** which holds context information, such as the position in the logging area where NTFS should start reading during a recovery.
    - The **Restart Area** holds two copies of its information, so recovery is still possible if one copy is damaged during the crash.
- The logging functionality is provided by the **log-file service**.
- Log-file service keeps track of the free space in the log file. If the free space gets too low, the log-file service queues pending transactions, and NTFS halts all new I/O operations
- After the in-progress operations complete, NTFS calls the cache manager to flush all data and then resets the log file and performs the queued transactions.

Case study : Windows File System - Security

- The security of an NTFS volume is derived from the Windows object model
- Each NTFS file references a security descriptor, which specifies the owner of the file, and an access-control list, which contains the access permissions granted or denied to each user or group listed.
- In normal operation, NTFS does not enforce permissions on traversal of directories in file path names.
- For compatibility with POSIX , these checks can be enabled.
- Prefix matching is an algorithm that looks up strings in a cache and finds the entry with the longest match—for example, an entry for \ home\sridatta would be a match for \home \sridatta \Desktop \Dynamic_AER.jpg
- The prefix-matching cache allows path-name traversal to begin much deeper in the tree, saving many steps. Enforcing traversal checks means that the user's access must be checked at each directory level.

Case study : Windows File System - Volume Management and Fault Tolerance

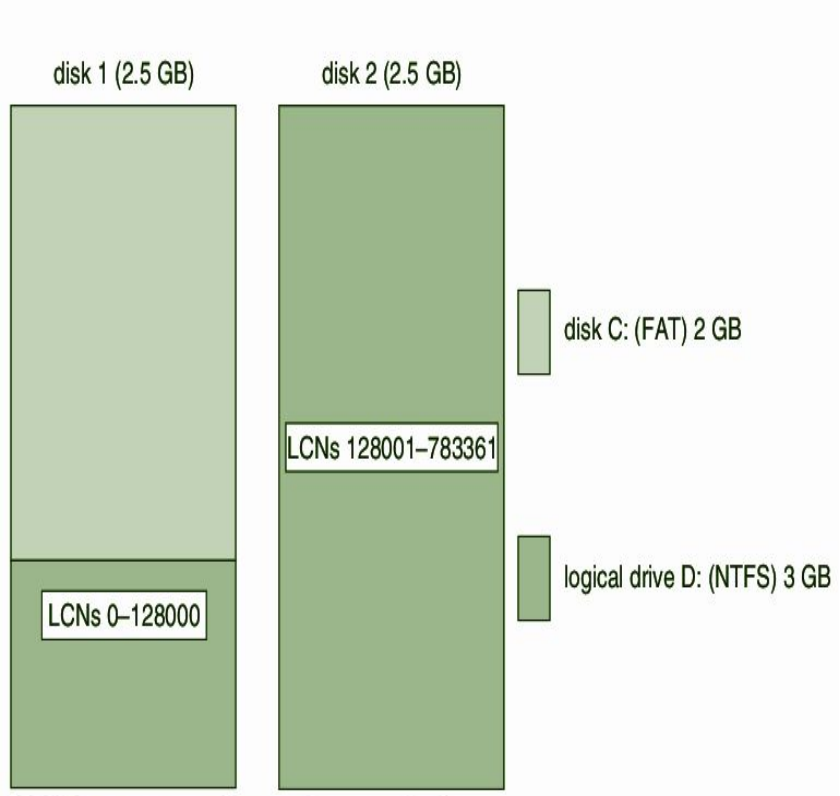
- **FtDisk** is the fault-tolerant disk driver for Windows.

- When installed, it provides several ways to combine multiple disk drives into one logical volume so as to improve **Performance**, **Capacity**, or **Reliability**

Case study : Windows File System - Volume Management and Fault Tolerance

Volume Sets and RAID Sets

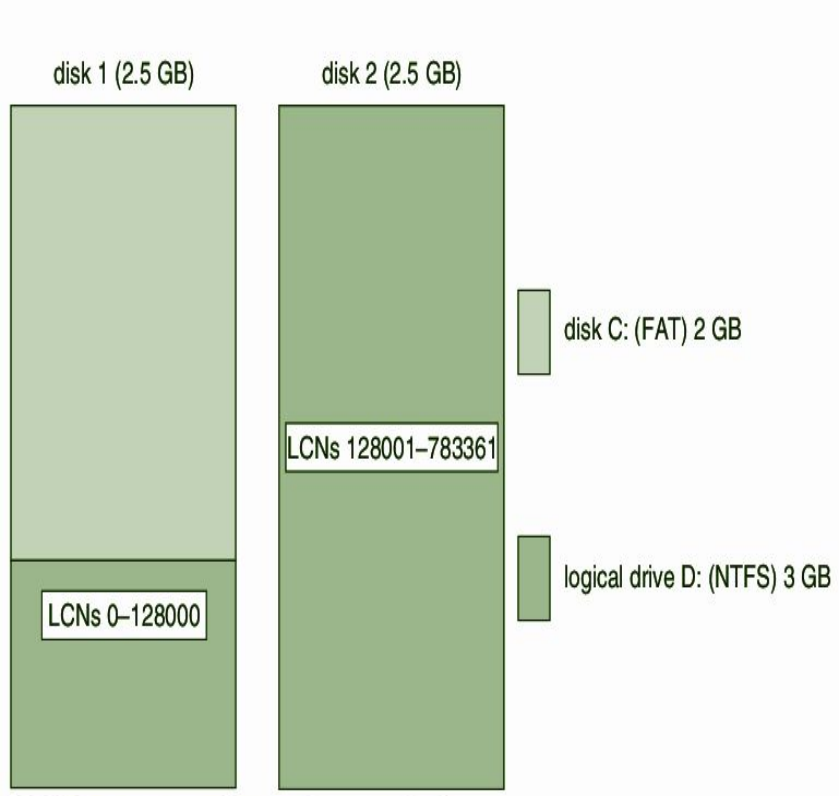
- One way to combine multiple disks is to concatenate them logically to form a large logical volume, as shown in Figure
- In Windows, this logical volume, called a **volume set**, can consist of up to 32 physical partitions.
- A volume set that contains an NTFS volume can be extended without disturbance of the data already stored in the file system.
- The bitmap metadata on the NTFS volume are simply extended to cover the newly added space.



Volume set on two drives

Volume Sets and RAID Sets

- Second way to combine multiple physical partitions is to interleave their blocks in round-robin fashion to form a stripe set.
- This scheme is also called RAID level 0, or **disk striping**.
- Windows also supports RAID level 5, stripe set with parity, and RAID level 1, mirroring.



Volume set on two drives

Sector Sparing and Cluster Remapping

- To deal with disk sectors that go bad, FtDisk uses a hardware technique called **Sector Sparing**, and NTFS uses a software technique called **Cluster Remapping**.
- **Sector sparing** is a hardware capability provided by many disk drives.
- When a disk drive is formatted, it creates a map from logical block numbers to good sectors on the disk.
- **Cluster Remapping** is a software technique performed by the file system.
- If a disk block goes bad, NTFS substitutes a different, unallocated block by changing any affected pointers in the MFT.
- NTFS also makes a note that the bad block should never be allocated to any file.

Sector Sparing and Cluster Remapping

- When a disk block goes bad, the usual outcome is a data loss.
- But sector sparing or cluster remapping can be combined with fault-tolerant volumes to mask the failure of a disk block.
- If a read fails, the system reconstructs the missing data by reading the mirror or by calculating the exclusive or parity in a stripe set with parity.
- The reconstructed data are stored in a new location that is obtained by sector sparing or cluster remapping.

Case study : Windows File System - Compression

- NTFS can perform data compression on individual files or on all data files in a directory.
- To compress a file, NTFS divides the file's data into compression units, which are blocks of 16 contiguous clusters.
- When a compression unit is written, a data-compression algorithm is applied.
- If the result fits into fewer than 16 clusters, the compressed version is stored.
- For sparse files or files that contain mostly zeros, NTFS uses another technique to save space. Clusters that contain only zeros because they have never been written are not actually allocated or stored on disk.
- Instead, gaps are left in the sequence of virtual-cluster numbers stored in the MFT entry for the file.
- When reading a file, if NTFS finds a gap in the virtual-cluster numbers, it just zero-fills that portion of the caller's buffer. This technique is also used by UNIX .

Case study : Windows File System - Compression

Mount Points, Symbolic Links, and Hard Links

- Mount points are a form of symbolic link specific to directories on NTFS that were introduced in Windows 2000.
- They provide a mechanism for organizing disk volumes that is more flexible than the use of global names like drive letters.
- A mount point is implemented as a **symbolic link** with associated data that contains the true volume name
- The links can be absolute or relative, can point to objects that do not exist, and can point to both files and directories even across volumes.
- NTFS also supports **hard links**, where a single file has an entry in more than one directory of the same volume.

Case study : Windows File System - Compression

Mount Points, Symbolic Links, and Hard Links

- A **hard link** acts as a copy (mirrored) of the selected file.
    - It accesses the data available in the original file.
    - If the earlier selected file is deleted, the hard link to the file will still contain the data of that file.
- A **Soft Link** also known as **Symbolic link** acts as a pointer or a reference to the file name.
    - It does not access the data available in the original file.
    - If the earlier file is deleted, the soft link will be pointing to a file that does not exist anymore.

Case study : Windows File System - Change Journal

- NTFS keeps a journal describing all changes that have been made to the file system.
- User-mode services can receive notifications of changes to the journal and then identify what files have changed by reading from the journal.
- The search indexer service uses the change journal to identify files that need to be re-indexed.
- The file-replication service uses it to identify files that need to be replicated across the network.

Case study : Windows File System - Volume Shadow Copies

- Windows implements the capability of bringing a volume to a known state and then creating a shadow copy that can be used to backup a consistent view of the volume.
- This technique is known as **Snapshots** in some other file systems.
- Making a shadow copy of a volume is a form of copy-on-write, where blocks modified after the shadow copy is created are stored in their original form in the copy.
- The server version of Windows uses shadow copies to efficiently maintain old versions of files stored on file servers

- Case study : Windows File System - 2

Course Syllabus Completed

for all the

Units (Unit 1, Unit 2, Unit 3, Unit 4, Unit 5)

Thank you

THANK YOU

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

nitin.pujari@pes.edu

For Course Deliverables by the Anchor Faculty click on www.pesuacademy.com