Example File Systems

Chapter 4: Section 4.5

MS-DOS File System

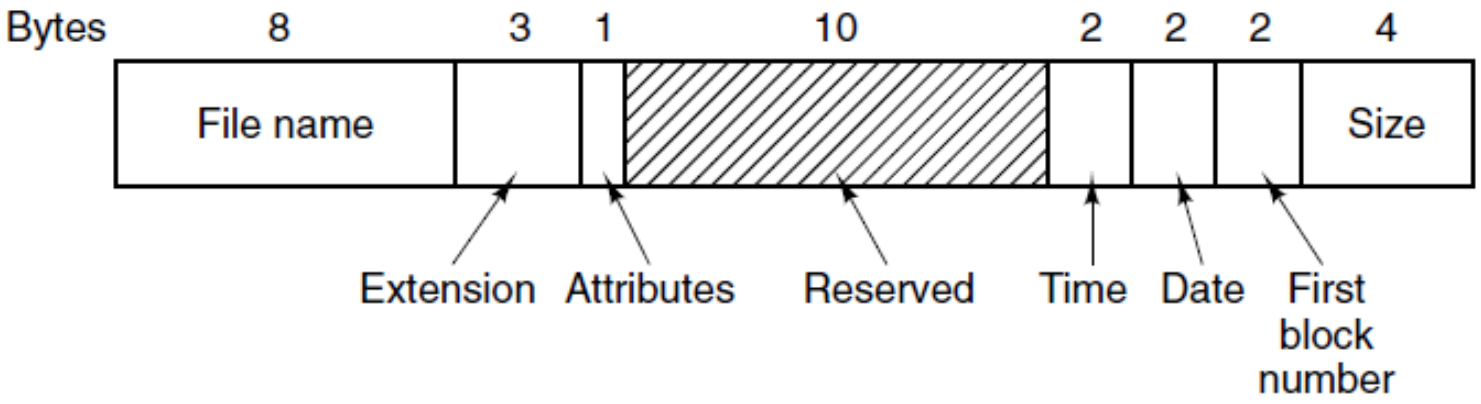
- First file system for IBM PCs.
- Main file system up through Windows 98 and Windows ME
  - Still supported in Windows 2000, Windows XP, Windows Vista.
- Extension (FAT-32) still widely used in digital cameras, mp3 players, thumb drives

MS-DOS File System

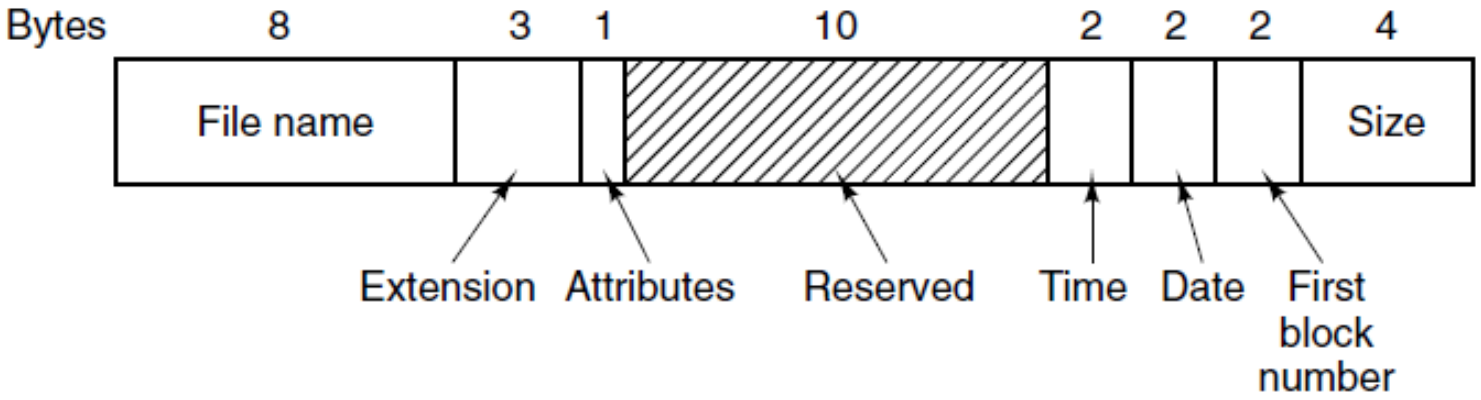
» More devices use FAT-32 than NTFS

» MS-DOS directories are variable sized.

» Each directory entry is fixed 32 bytes.



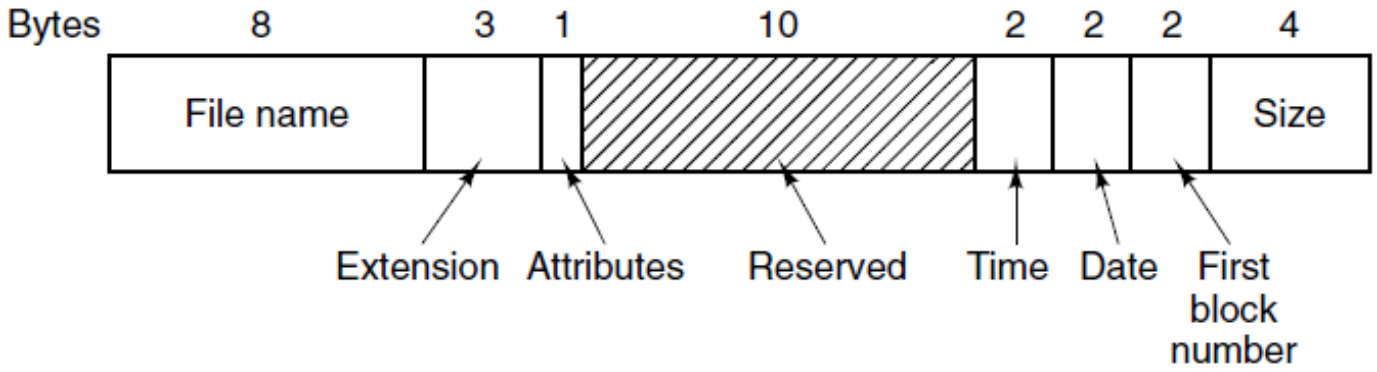
MS-DOS File System



» File names max size is 8+3, left justified and padded with 0 on the right if smaller.

» Attributes for read/write/hidden/archive/system.

MS-DOS File System



» Time stores creation time.

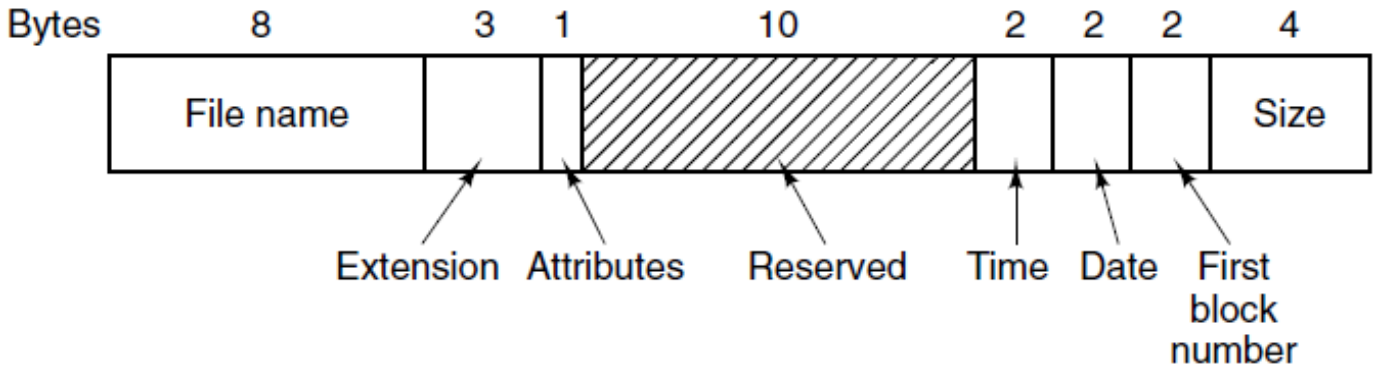
  » Accurate to +/- 2 seconds since it's stored in a 2 byte field.

    » Only 65,535 unique values.

    » A day has 86,400 seconds

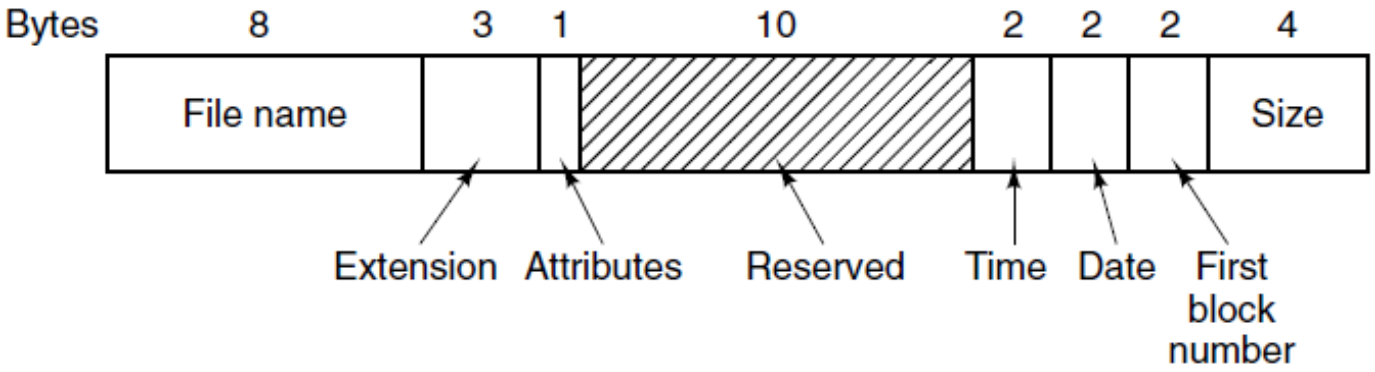
    » seconds (5bits), minutes (6bits) hours (5bits)

MS-DOS File System



- Date stores the date in days.
  - Day (5bits) , Month (4bits), Year (7bits)
  - Year expressed in years since 1980.
  - Max year is 2107.
  - Y2108 problem.

MS-DOS File System

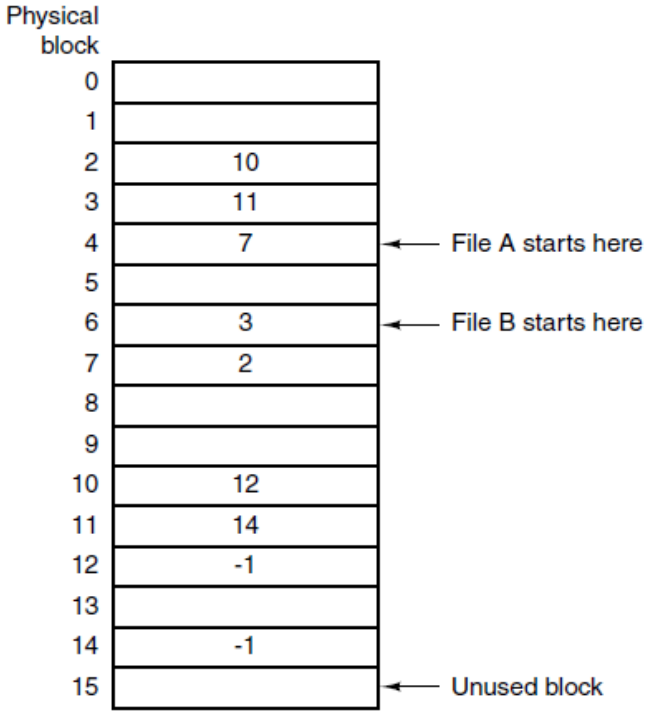


- Size is stored as 32-bit number. Theoretical max file size of 4 GB
  - Other limitations limit this to 2 GB.
  - 10 reserved bits unused.

MS-DOS File System

- MS-DOS tracks blocks via File Allocation Table (FAT)
- The directory entry contains the number of the first block.
  - Used as an index into the 64K FAT in main memory.

MS-DOS File System



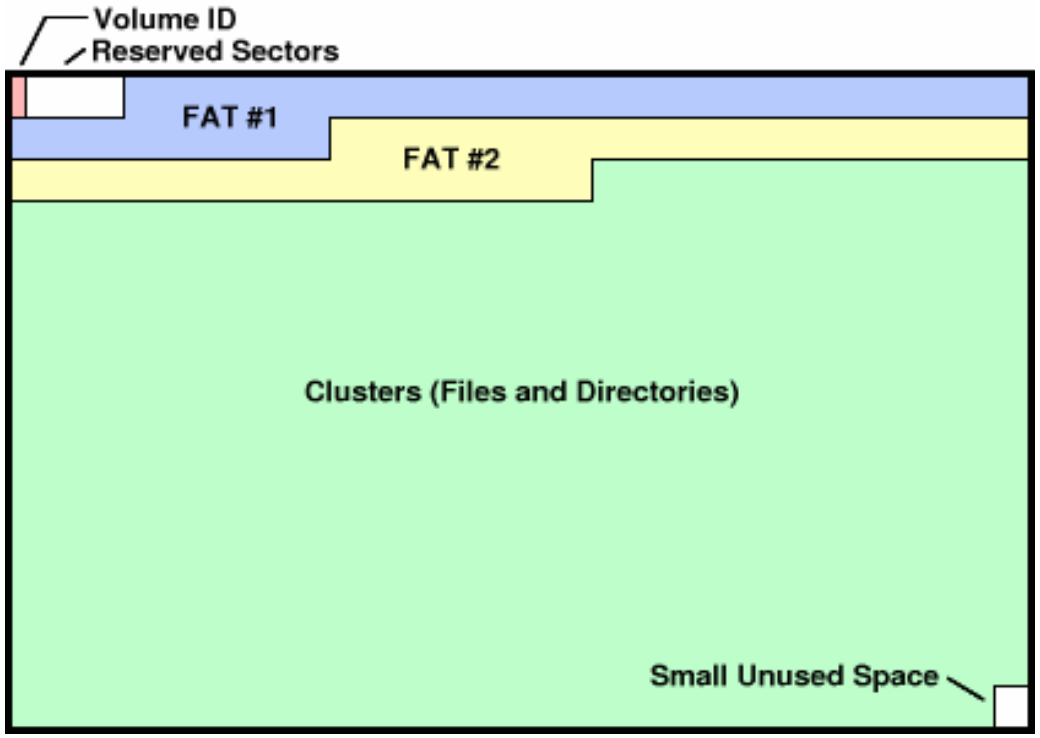
Linked list allocation using a file allocation table in main memory.

No bitmap or free list required.

MS-DOS File System

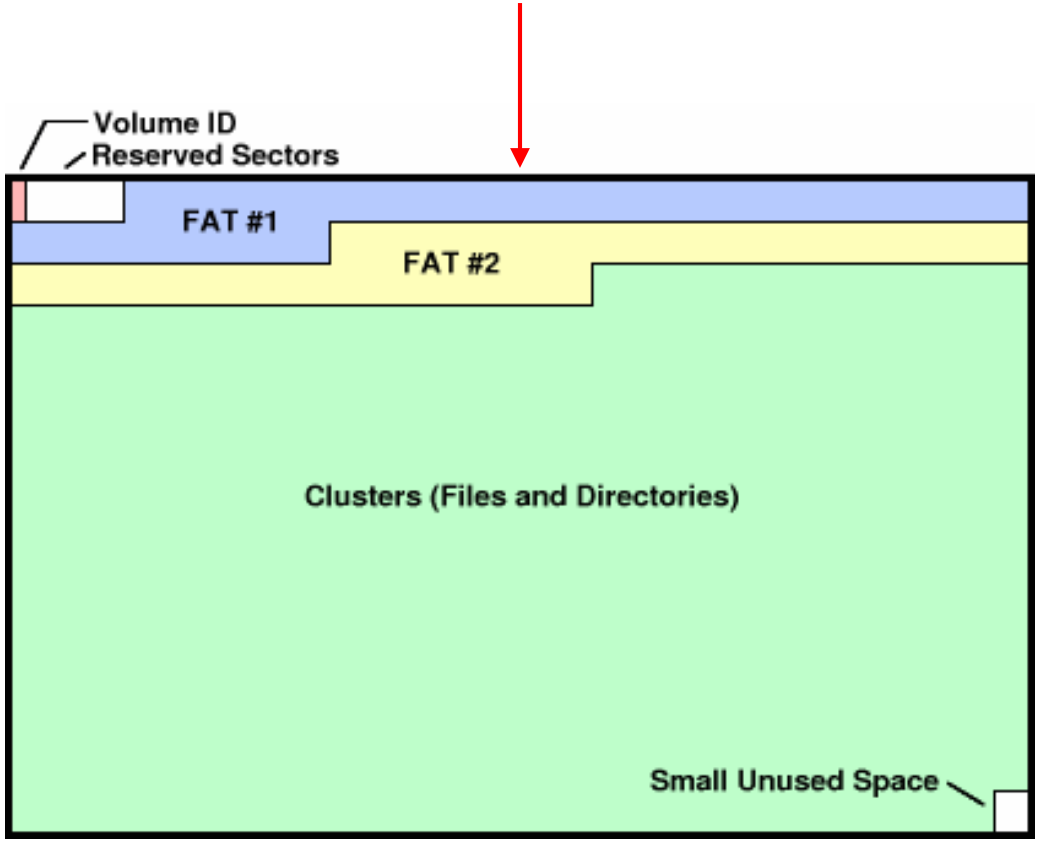
- Three versions of FAT.
  - FAT-12, FAT-16, FAT-32
- FAT-32 misnomer. Only 28 bits are actually used.
- exFAT - Microsoft introduced for large removable devices.
  - Apple licensed so Windows to OS X transfers of exFAT can occur
  - Spec not public

FAT32 Layout



FAT32 Layout

Each FAT has 1 32-bit word for every cluster. Each entry is the logical block of the next block in the file.



MS-DOS File System

- FAT disk blocks are multiples of 512 bytes.
  - Block size called cluster size
- Can vary per partition

Block size	FAT-12	FAT-16	FAT-32
0.5 KB	2 MB		
1 KB	4 MB		
2 KB	8 MB	128 MB	
4 KB	16 MB	256 MB	1 TB
8 KB		512 MB	2 TB
16 KB		1024 MB	2 TB
32 KB		2048 MB	2 TB

MS-DOS File System

- When hard disks arrived 2MB partitions were too small.
- MS allowed additional blocks sizes of 1KB, 2KB, and 4KB
  - Preserved 12 bit FAT table
  - Allowed disk partitions up to 16 MB
- 4 disk partitions per drive so could support up to 64 MB disks

MS-DOS File System

- FAT-16 with 16 bit pointers introduced
  - Additional block sizes of 8KB, 16KB, 32 KB.
- FAT-16 table occupied 128KB of main memory.
- 2GB largest partition.
  - 64K entries of 32KB each.
- 8GB disk max. (4 partitions * 2 GB)

MS-DOS File System

- Second release of Windows 95 introduced the FAT-32 file system with 28-bit disk addresses.
- Theoretical the partitions could be 2^{28} x 2^{15} bytes
- Actual limit is 2TB (2048 GB)
  - Internally tracks partition sizes with a 32 bit number. 2^{9} x $2^{32} = 2$ TB
- Max 8GB in single partition
- Can use 4KB blocks for large partitions.

MS-DOS File System

- Second release of Windows 95 introduced the FAT-32 file system with 28-bit disk addresses.
- Theoretical the partitions could be 2^{28} x 2^{15} bytes
- Actual limit is 2TB (2048 GB)
  - Internally tracks partition sizes with a 32 bit number. 2^{9} x $2^{32} = 2$ TB
- Max 8GB in single partition
- Can use 4KB blocks for large partitions.

CD-ROM File System

- Simple since designed for write-once media
- No provision for tracking free blocks since blocks can't be freed or added to once manufactured.
- CD-R (CD Recordable) introduced the possibility to add files.
    - Appended to the end of the file system.
    - All free space is one contiguous chunk at the end of the file system.

ISO 9660 File System

- Adopted as a standard in 1998.
- Virtually every CD on the market is compatible.
- Goal was make every CD ROM readable on every computer independent of byte ordering and OS used.
    - Caused limitations to be placed on the system.

ISO 9660 File System

- CD-ROM do not have concentric cylinders.
  - Single continuous spiral containing the bits in a linear sequence.
  - Logical seeks across the spiral are still possible..
  - Bits along the spiral are divided into logical blocks (also called logical sectors) of 2352 bytes.
    - Some are preamble, ECC and other overhead.
  - Payload of each block is 2048 bytes

ISO 9660 File System

- When used for music CDs have leadins, leadoffs and inter track gaps. Not used in data formats.
- Block position sometimes quoted in minutes and seconds
  - 1 sec = 75 blocks.
- Each CD begins with 16 blocks that are undermined by the standard and can be used to hold a bootstrap program.

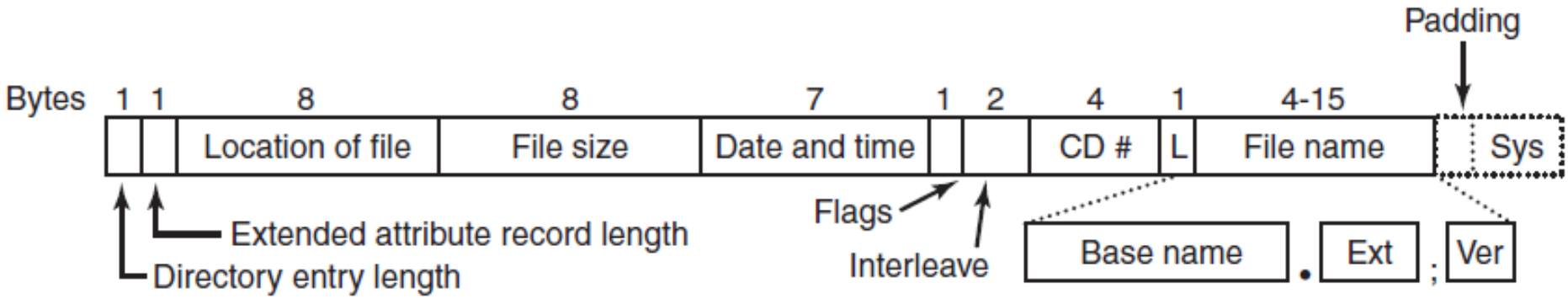
ISO 9660 File System

- Following is the primary volume descriptor
  - System Identifier (32 bytes)
  - Volume Identifier (32 bytes)
  - Publisher Identifier (128 bytes)
  - Data Preparer Identifier (128 bytes)
- All must be uppercase and only a few punctuation supported.

ISO 9660 File System

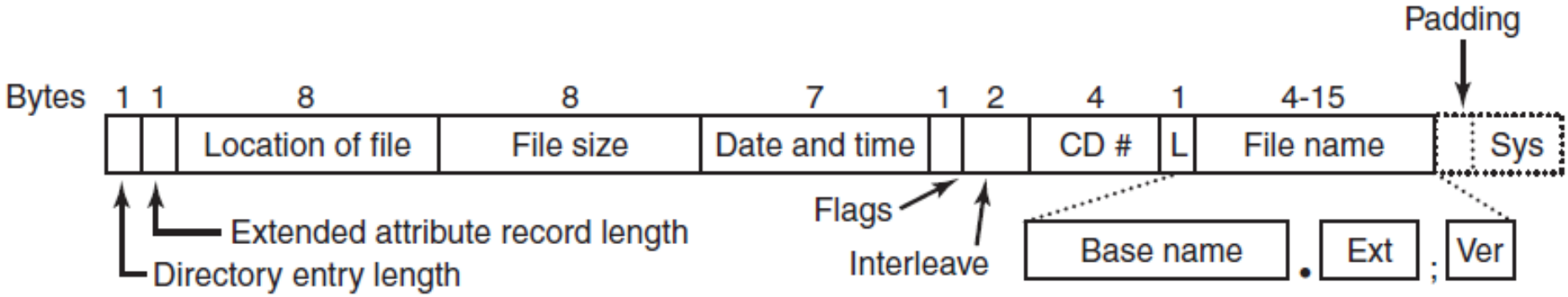
- Also contains the name of three files which contain:
  - Abstract
  - Copyright Notice
  - Bibliographic Information
- Also holds block size (normally 2048 but 4KB, 8KB and higher powers of 2 can be supported.)
- Contains directory entry for the root directory.

ISO 9660 File System



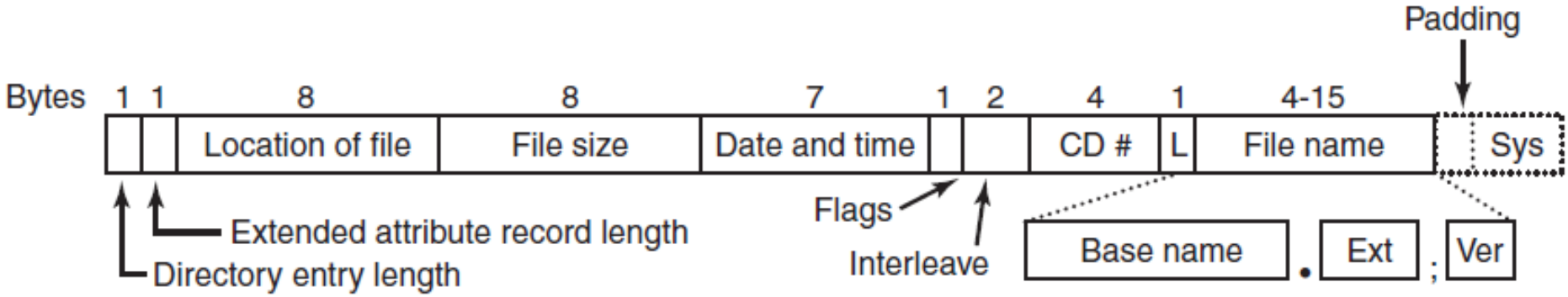
- Variable number of entries
  - Last contains marker signifying end
  - Each entry encoded in ASCII or binary.
    - Binary encoded twice. Little endien and big endien

ISO 9660 File System



- Directory entires may have extended attributes
- Date and time stored in byte each for day, month, year
  - Years started at 1900. Y2156 problem.

ISO 9660 File System



- No limit to number of entries in a directory.
- There is a limit of eight to nesting.
  - Arbitrarily set to make some implementations easier.

ISO 9660 File System

- Three levels
  - Level 1:
    - 8+3 file names
    - All files contiguous
    - 8 character file names
  - Level 2:
    - Up to 31 character file and directory names
  - Level 3:
    - Same limits as level 2 bt files do not have to be contiguous.

ZFS

Zetabyte FileSystem

ZFS

- **Hardware Agnostic**
  Runs on servers with directly attached storage. No fancy RAID cards or SAN.

- **Pooled Storage**
  No volumes. All capacity is available to all file systems.

- **Transactional**
  Always consistent on disk. No fsck. Ever.

- **Data Integrity**
  Detects silent data corruption. Automatic corrects detected errors.

- **Simple Administration**

ZFS

- Storage Pools
  - Constructed of files, partitions, or entire disks
    - Does for storage what VM did for memory
  - stripe, mirror(RAID1), raidz(RAID5), raidz2, raidz3
    - Hot spares
- Data Integrity
  - Checksums
  - Online “scrub”
    - fsck is offline and only check metadata
    - scrub once per week for cheap disks or per month for enterprise disks
- Capacity
  - 128-bit file system

ZFS

- Copy-on-write
  - Modified data is in a new block
- Snapshots & Clones & Rollbacks
  - Fast creation
  - Space efficiency
  - Clones are writeable snapshots
- Dynamic striping
  - Across all devices to maximize throughput
- Compression
  - lzjb, gzip-*

ZFS

- Variable block size
  - Data compression (CPU-bound vs. I/O-bound)
- Adaptive endianness
  - Sparc(Big-endian) & others(Little-endian)
- Deduplication
  - For space efficiency
  - 2GB ram / 1T disk
- Encryption
  - ZFS Pool Version 30

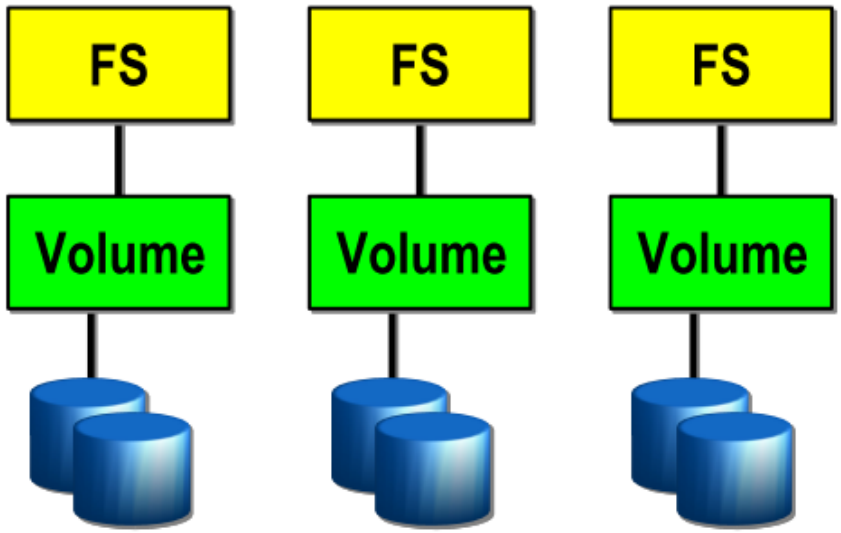
ZFS

- Platforms
  - Solaris 10 / 11
  - OpenSolaris / OpenIndiana
  - FreeBSD
    - 8.2-R: v15
    - 8.2-S, 9.0-, 10.0-C: v28
  - FreeNAS
  - GNU/kFreeBSD
  - NetBSD
  - Mac X OS
  - Linux / Linux FUSE / Kernel Module

FS/Volume model vs. ZFS

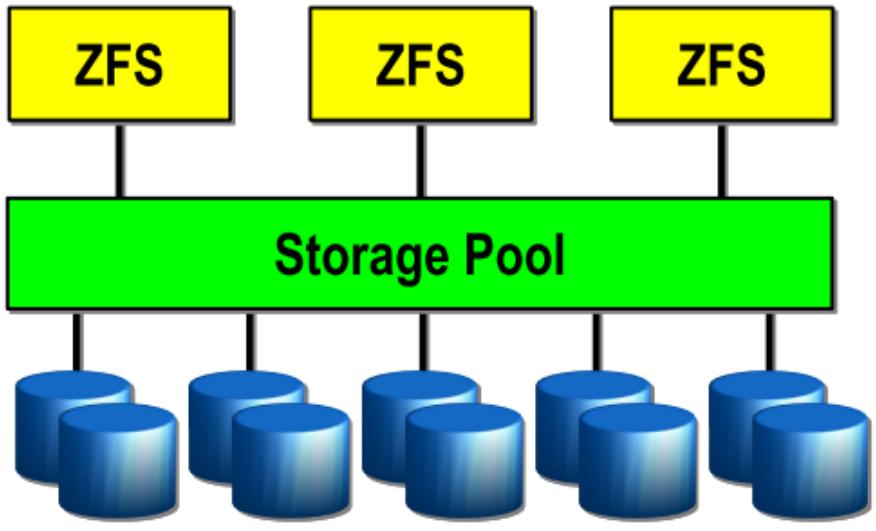
Traditional Volumes

- Abstraction: virtual disk
- Partition/volume for each FS
- Grow/shrink by hand
- Each FS has limited bandwidth
- Storage is fragmented, stranded



ZFS Pooled Storage

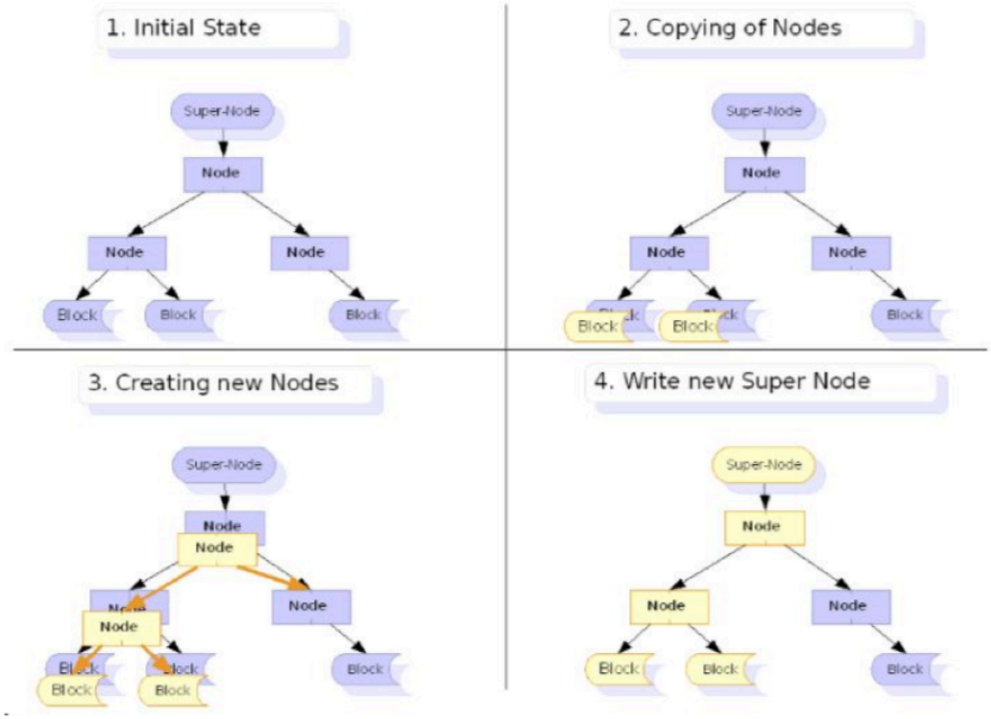
- Abstraction: malloc/free
- No partitions to manage
- Grow/shrink automatically
- All bandwidth always available
- All storage in the pool is shared



Copy-On-Write

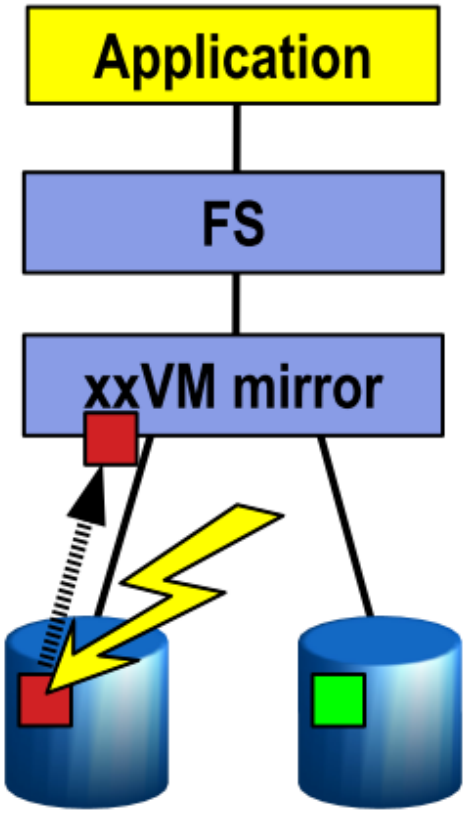
Copy-On-Write Transactions

- **Never overwrite block when modifying data:**
  - Write modified data on new block
  - Update pointers to new block
- **Free versioning / snapshots with old blocks & pointers**
- **Increased disk fragmentation**
  - Mitigating with read/write caching features

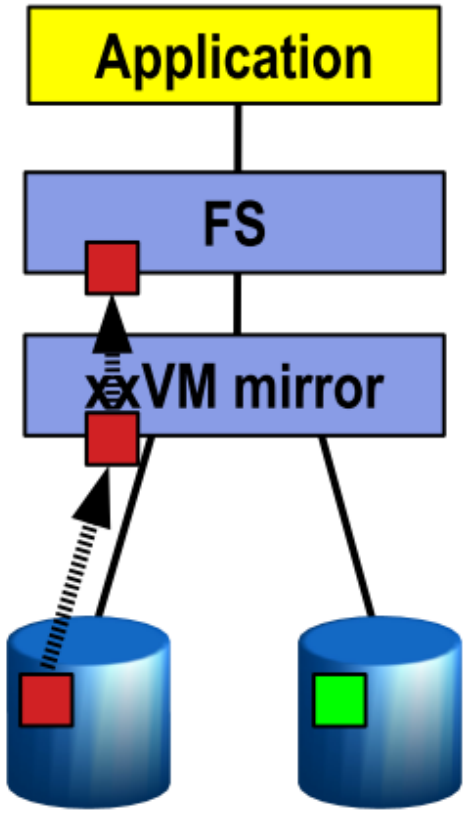


Traditional Mirroring

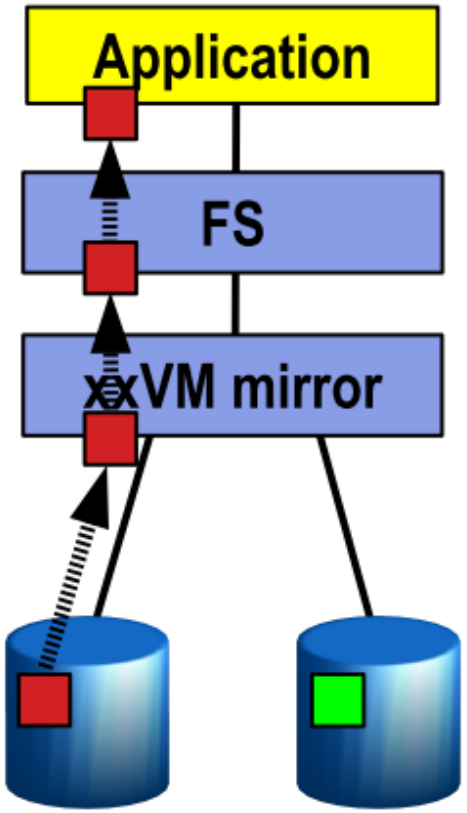
1. Application issues a read. Mirror reads the first disk, which has a corrupt block. It can't tell.



2. Volume manager passes bad block up to filesystem. If it's a metadata block, the filesystem panics. If not...

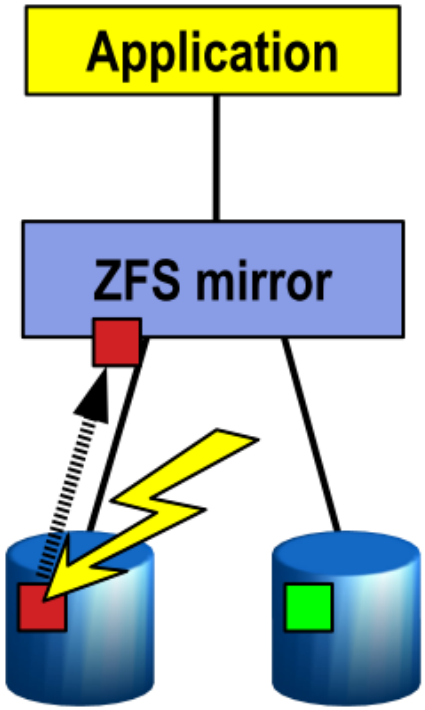


3. Filesystem returns bad data to the application.

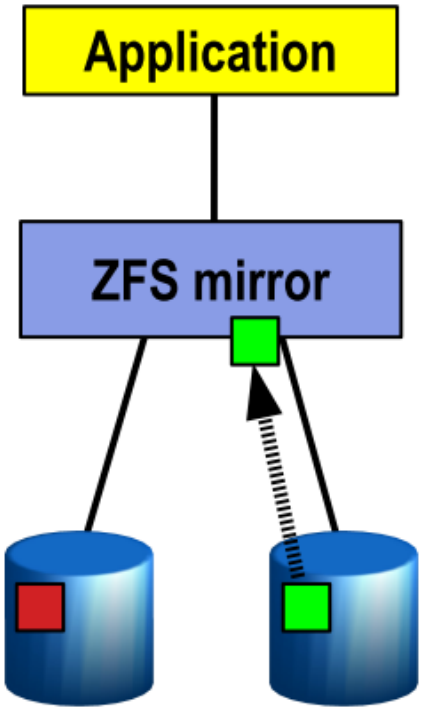


Self-Healing in ZFS

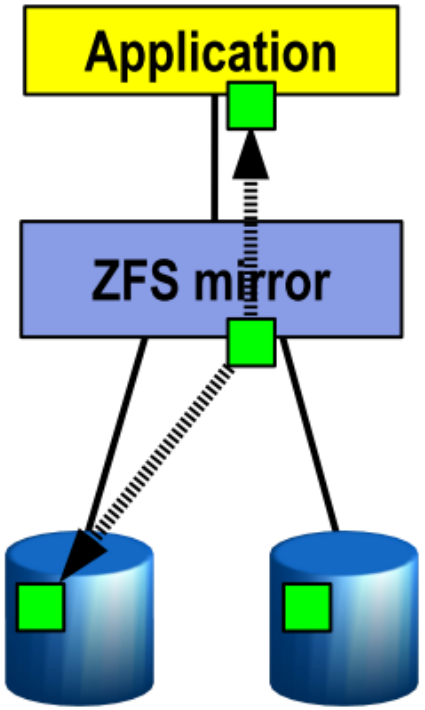
1. Application issues a read. ZFS mirror tries the first disk. Checksum reveals that the block is corrupt on disk.



2. ZFS tries the second disk. Checksum indicates that the block is good.



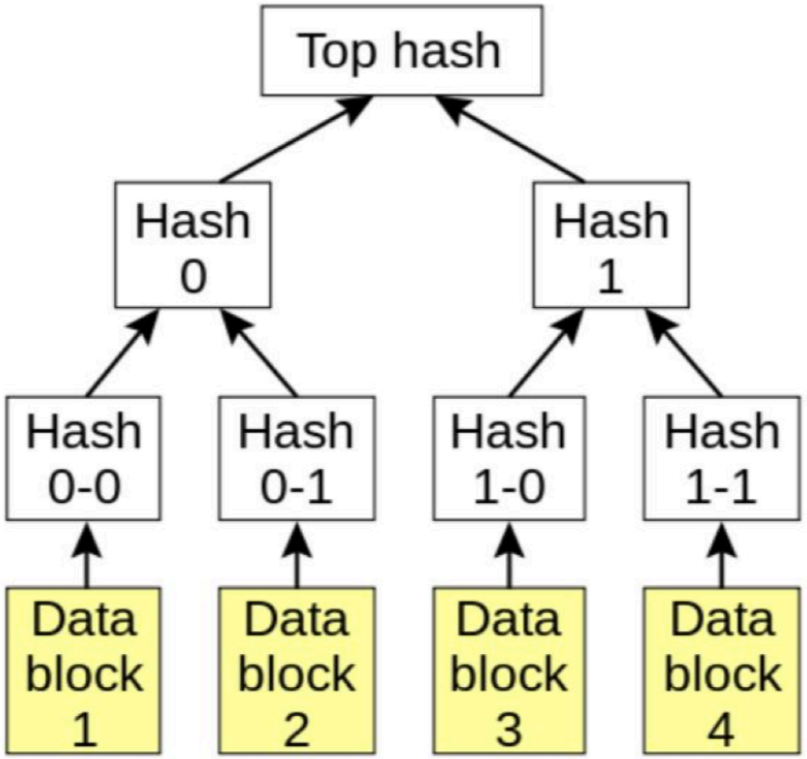
3. ZFS returns known good data to the application and repairs the damaged block.



Self-Healing in ZFS

Bit-Rot Detection

- **Merkle Tree (Hash Tree)**
  - Each block of data is hashed
  - Parent hashes are computed from child hashes
  - Corruption/changes to data is detected when traversing the tree

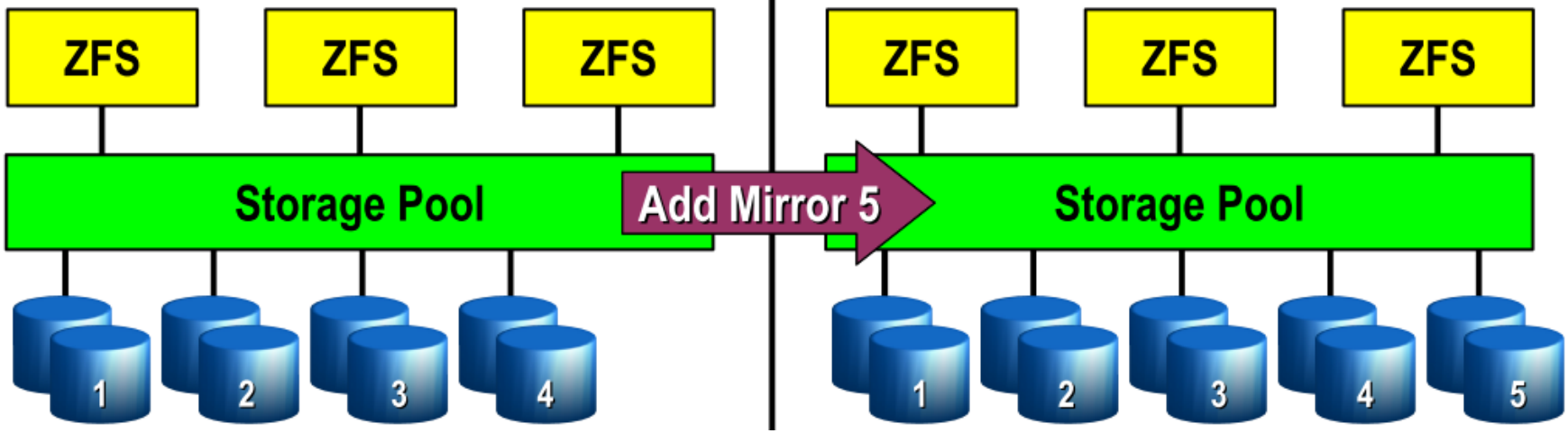


Dynamic Striping

- Automatically distributes load across all devices

- Writes: striped across all four mirrors
- Reads: wherever the data was written
- Block allocation policy considers:
  - Capacity
  - Performance (latency, BW)
  - Health (degraded mirrors)

- Writes: striped across all five mirrors
- Reads: wherever the data was written
- No need to migrate existing data
  - Old data striped across 1-4
  - New data striped across 1-5
  - COW gently reallocates old data



Miscellaneous Features

- **Compression**
  lz4 saves space, increases performance, and should be enabled.

- **Scalability**
  zpools are scalable by adding more devices. Workload is automatically distributed.

- **Snapshots (.zfs/snapshot in root of each file system)**
  - Instantaneous creation, unlimited number
  - No additional space used
  - Ideal uses: incremental backups/self-served restores, sharing read-only data