Doing Storage Better
Exploring the best and worst options

Art S. Kagel
ASK Database Management

WAIUG December 8, 2022

How you do IO matters!

- Prior to v11.10 IDS had to open Cooked Device and File chunks with the o_sync flag enabled in order to insure that data was safely on disk when a write system call returned
- Data was being copied from IDS buffers to OS buffers before being written to disk
- Synchronous writes are slow
- This caused Cooked Device chunks to be ~10-15% slower than RAW Device chunks
- Filesystem chunks were 15-25% slower than RAW

How you do IO matters!

- IDS 11.10 began support for opening filesystem based chunk files with the new o_direct (aka DIRECT_IO) flag which by-passes the OS Cache and writes directly to disk
- Only supported for Filesystem chunks – not Cooked Device chunks
- Makes Filesystem chunks perform ~5% slower than RAW Device chunks when properly configured

How you do IO matters!

- IDS v11.50 added support for the AIX Concurrent IO model using JFS2 filesystems
- On AIX with o_cio support configured for a JFS2 filesystem and the filesystem properly tuned and mounted, Filesystem chunks are only about 3-5% slower than RAW Device chunks
- IDS v14.10 added the ability to use o_direct with temp dbspaces

Where you do IO matters!

- Filesystems come in three flavors:
  - “Normal”
  - “Light weight”
  - Journaling

Where you do IO matters!

- General filesystem information -
  - IDS pre-allocates chunks so fragmentation of the initial chunk will depend on the state of the filesystem at the time you create the chunk (except some journaled FS)
  - Marking a chunk expandable increases the likelihood that the new allocations appended to the chunk will not be contiguous
  - You have no control over the layout of the chunk's disk allocations within the filesystem
  - Expandable filesystems exacerbate the problem

Where you do IO matters!

- “Normal” filesystems
  - Average overhead, though different filesystems behave differently
  - Most UNIX filesystems attempt to allocate a new file from contiguous disk, but no guarantees for larger files once files have been created and destroyed on the filesystem over time
  - Best to use a brand new filesystem for chunks!

Where you do IO matters!

- “Light weight” filesystems
  - Designed by OS and SAN vendors for high speed applications
  - Supposed to be lower overhead than “normal” filesystems
  - The primary advantage is they are faster to open and close files
  - Anecdotal evidence indicates that they are not noticeably better for database system storage since typically databases open their files once on startup and leave them open

Where you do IO matters!

- Journaling filesystems
  - Adds a form of logging to the filesystem to reduce recovery time and limit data loss if the system crashes with unwritten data in cache.
  - Do not need periodic “cleaning”
  - Many different versions:
    - JFS/JFS2/OpenJFS – IBM developed for AIX and released as open source
    - ZFS – Sun developed for Solaris and released as open source
    - EXT3 & EXT4 – Linux developed
    - VxFS - Veritas developed
    - XFS – Silicon Graphics developed for IRIX

Where you do IO matters!

- JFS/JFS2/OpenJFS
  - Journals filesystem meta-data only (not file contents)
  - Serialized writes to maintain data consistency unless Concurrent_IO is enabled
  - Uses variable length extents to build large files
  - Maps/locates extents using a btree index in the inode
  - JFS locks the FS allocation groups during file expansion to improve contiguous allocation. This can block the expansion of other files in that allocation group
  - Fairly low overhead
  - Fairly safe

Where you do IO matters!

- ZFS – Sun developed for Solaris and released as open source
  - Uses copy-on-write transaction model – rewrites are made to a different physical location than data came from and the new block replaces the old one in the meta-data and are later “cleaned” by background threads. This causes chunks to become more and more non-contiguous over time
  - IDS pages are substantially smaller than ZFS pages. That means many rewrites of the same ZFS page and many relocations that have to be cleaned up in the background which affects performance

Where you do IO matters!

- ZFS – Sun developed for Solaris and released as open source
  - Dirty block cleanup eats into IO bandwidth and fights applications for head positioning on spindle disks
  - COW can cause SSDs to age prematurely
  - ZFS maintains two to three checksums for every data and meta-data block modifying each up the meta-data tree for every write
  - High overhead
  - Good safety

Where you do IO matters!

- EXT3 & EXT4 – Linux developed
  - Essentially EXT2 with journaling added on.
  - Data and metadata journaled
  - Copy-on-write data rewrites – causes chunks to become increasingly less and less contiguous over time
  - COW can cause SSDs to age prematurely

Where you do IO matters!

- EXT3 & EXT4 – Linux developed
  - EXT4 (& EXT3 with write-back enabled) writes single entry metadata journal entries BEFORE the data block it maps. Can CAUSE corruption if the system crashes before the updated data is written

    Linus Torvalds says: “Whoever came up with (*EXT4's write back policy*) was a moron. No ifs, buts, or maybes about it.”
  - EXT4 writes out dirty cache only every 2 minutes (EXT2 & EXT3 do so every 5 seconds)
  - Low safety
  - Low performance
  - Prefer EXT3 with journaling disabled

Where you do IO matters!

- VxFS is proprietary, so skip it
- XFS
  - Meta-data only journaling
  - Write journal before data
  - Dual entry journaling to permit recovery if the modified data is never written
  - Low overhead
  - Good safety

What's behind the scenes matters!

Let's talk about RAID Levels!

What's behind the scenes matters!

RAID0 – Striping only
RAID1 – Mirrored drives only

Combinations or RAID0 & RAID1

RAID01 – Mirror two stripe sets
RAID10 – Stripe two or more mirror sets

What's behind the scenes matters!

- RAID5! - One extra drive per array to allow for parity blocks. Rotating parity location.
- RAID6! - Two extra drives per array to allow for double parity blocks. Rotating parity location
- RAID51! - One parity drive per array and mirror the entire array on another. <LOL!>
- RAID61! - Two parity drives per array and mirror the entire array on another. <ROTFL!>
- RAID50! - Two or more RAID5 arrays connected ala RAID0. Less data per parity block. <False hope!>
- RAID60! - Two or more RAID6 arrays connected ala RAID0. Even less data per parity block. <More expansive false hope!>

What's behind the scenes matters!

NO RAID5 !!!!!!!!!!!!!!!
NO RAID6 !!!!!!!!!!!!!!!
NO RAIDZ !!!!!!!!!!!!!!!
NO RAID51 !!!!!!!!!!!!!!!
NO RAID61 !!!!!!!!!!!!!!!
NO RAID50 !!!!!!!!!!!!!!!
NO RAID60 !!!!!!!!!!!!!!!

What's behind the scenes matters!

- Finally I am supported by the industry! The following points are now recognized by industry studies:
  - Possibility of a second drive failure is 4x more likely in practice than statistics predict!
    - The safety of RAID5 is predicated on the drive failure rate being low!
  - Server-grade drives have the same failure rates as consumer-grade drives!
    - What are you paying 2-3X the price for anyway?
  - Atomic writes to the multiple drives in an array are not guaranteed!
    - What happens if all of the data and parity drives in a RAID5 array are not both written atomically?

http://en.wikipedia.org/wiki/RAID#Problems_with_RAID

What's behind the scenes matters!

- Finally I am supported by the industry! The following are now recognized by industry studies:
  - Larger drives take longer to rebuild increasing risk of multiple drive failure over conditions in the past when drives were smaller.
  - A recent study concluded that drives over 1TB are statistically likely to suffer from multiple bit dropouts. Number of bits on the drive exceeds the bit failure rate!
  - SSD Flash units suffer from 'bit rot' and cosmic ray damage just like mechanical/magnetic disks. More so if they are frequently written to.
  - All in all, the error rates as observed by a CERN study on silent corruption, are far higher than the official rate of one in every 10^16 bits (observed error rates of about one in 10^7 bits ie 1 out of about 1 in every 1,000,000 bits)

http://en.wikipedia.org/wiki/RAID#Problems_with_RAID

What's behind the scenes matters!

- From a Dell web page:

Dell recommends not using RAID 5 for any business-critical data.

RAID 5 carries higher risks of encountering an uncorrectable drive error during a rebuild, and therefore does not offer optimal data protection.

What's behind the scenes matters!

- Scientists at CERN beat the hell out of 1.5PB of data on RAID5 and noticed data corruption so they studied it as only physicists can. Their report said:

The RAID controllers don't check the 'parity' when reading data from RAID 5 file systems. In principle the RAID controller should report problems on the disk level to the OS, but this seems not always to be the case.

http://indico.cern.ch/getFile.py/access?contribId=3&sessionId=0&resId=1&materialId=paper&confId=13797

Cern tested: A <program> was developed <that> writes a ~2 GB file containing special bit patterns and <then> reads the file back and compares the patterns. This program was deployed on more than 3000 nodes...and run every 2 hours. <Five weeks> of running on 3000 nodes revealed 500 errors on 100 nodes.

What's behind the scenes matters!

Cern found:

- 80% of their errors were traceable to disk firmware bugs
- 10% of errors traced to memory card incompatibility with system boards
- RAID5 could not correct any of these errors nor the remaining 10% due to bit rot (partial media failure and cosmic ray damage – note that Cern is very much underground!).

What's behind the scenes matters!

ONLY USE RAID10. Why?

- Safer from bit rot. Does not copy garbage from drive to drive.
- At least 75% less susceptible to data loss from second drive failure.
- 80% faster recovery time (further reduces 2nd drive failure risk).
- Performance degrades <10% during recovery
  - 6.25% for a 5 pair array versus 80% for a six drive RAID5 array
- Up to 200% higher peak read performance over RAID5. (Cern verified)
- Sustainable 100% increase in write performance versus RAID5 without adding huge and expensive cache memory on the SAN.
- Mirroring each drive from a different drive lot reduces the danger from hardware and firmware bugs in the drives causing failure of both sides of a mirrored pair. (This was the source of 80% of Cern's data corruption!)

What's behind the scenes matters!

A paper presented at FAST in 2007 by Google reveals that the probability of losing data during recovery from unrecoverable errors (UREs) alone on a <RAID5> array of 8 - 8TB SATA drives is over 98%.

That means that you have about a 1.1% chance of recovering your data before a corrupted sector makes one or more blocks completely lost causing the entire array to become unusable.

Questions about RAID5?!?

See these websites for more information:

www.askdbmgt.com/why-raid5-should-be-avoided-at-all-costs.html

www.baarf.com

Questions ?!?

Doing Storage Better

Art S. Kagel

art.kagel@gmail.com