Monday, July 13. 2009

The Register about ZFS deduplication

Even the Register reports about Deduplication in ZFS. I´m asking myself, if the people at the Register read my blog, as i´ve talked about that a few days ago.

Fun aside: Synchronous Dedupe is the only sensible way to do dedupe data as you would need to provide the storage for undeduped data otherwise until the system gets to the point where the data gets deduplicated. Dependent on the frequency of dedupe runs, this could be a vast amount of storage. On the other side, synchronous dedupe is dependent of a fast mechanism to detect duplicates. The checksumming feature of ZFS looks like a good way to do this, as it capable to use various hashing algorithms. When the probability of collision is less than the probability of reading wrong data from disks it should suffice just to check the checksums instead of checking the complete block.

Posted by Joerg Moellenkamp in English, Security at 14:34

still waiting for the day when that checksum-thing thinks your pr0n is your grandma's last birthday party video... should be quite fun when you play it in front of your whole family
Anonymous on Jul 13 2009, 19:56

Well ... the probability of such an occussion would be similar to the chance that spontaneous bit rot results in a HD porn video
Anonymous on Jul 13 2009, 20:21

let me quote smth from zfs-discuss: "fletcher2 is notoriously collision prone on real data sets. It is meant to be fast at the expense of collisions. This issue can show much more dedup possible than really exists on large datasets."

I think today's consumer 1TB+ drives should count as "large datasets"

http://mail.opensolaris.org/pipermail/zfs-discuss/2008-July/019700.html
Anonymous on Jul 13 2009, 20:32

I knew that you would cite this mail right in the moment, i submitted my comment. You are really predictable

Let me quote the man page for zfs

"checksum=on | off | fletcher2,| fletcher4 | sha256"

Especially the last one. It's already there. Right here, right now. The mechanism is pluggable. Thus you could implement Skein or MD6 or whatever you want.
Anonymous on Jul 13 2009, 21:36

fine, then why it's not on by default... maybe because

"Fletcher2 checksum (the default) has been observed to consume roughly 1Ghz of a CPU when checksumming 500 MByte per second."
-- http://www.solarisinternals.com/wiki/index.php/ZFS_Evil_Tuning_Guide

and flecher2 is a really simple checksum algorithm, so I guess sha256 needs a lot more cpu power... and we'll hit 500MB/s+ as soon as SATA makes the jump to 6Gbit/s

ah well, I think this dedup thing is nothing for me
Anonymous on Jul 13 2009, 22:44

Asynchronous dedup has the advantage of less latency being introduced to write operations.

The dedup can be completed in a bulk manner sometime after the data is initially written; I would think something along the lines of "zpool scrub".

So long as an ample reserve of spare disk space is available, as a temporary reserve.

Earlier dedupe operations themselves may have provided just that space.

Async dedup is sensible enough for commercial storage vendors to have implemented dedup primarily that way.

There are tradeoffs between synchronous and asynchronous, and one size definitely doesn't fit all.
Anonymous on Jul 14 2009, 04:23

Today you have vast amount of cycles for such tasks. Anyway, i don´t think you would use deduplication on your database working disks, you would use it on archives, backups and something like this. So it would be feasible to use sha256 or something similar for your archive dedup disks and fletcher2 for your high speed needs.

The point is: When the probability of an hash collision is less than the probability of an undetected bit-error that leads to the assumed equality of two blocks, then it could be feasible to omit the bit-wise comparison. You have to take one thing into consideration: This bit-wise compare takes an even more precious resource from your system. IOPS. For each check you have to jump away from your current location and read the other block.

It about probabilities and as long as you use filesystems without any checksums for data, this whole discussion is funny, as the probablity of corrupted data just because of the UBER of your harddisk is vastly higher than the probability of a hash collision.
Anonymous on Jul 14 2009, 08:00