

Wednesday, April 2, 2008

Dedupalicious

Deduplication is one of the big hype topics in the storage world today. Deduplication is about finding blocks or files on a storage, that exist multiple times with the same content. For example one of this unfunny joke presentation mailed to a big mail alias, which got saved on 50% of the desktops. Deduplication find this blocks or files and leaves only one to the disk, replacing all others with a pointer to the remaining copy. The basic idea: Saving capacity by reducing redundancy.

ZFS may have such an functionality in the future. Eric Kustarz writes in [how dedupalicious is your pool?](#) about some basics of such a functionality. The checksums integrated in ZFS may be of great use for Dedup, albeit you would use a cryptographically really strong hash algorithm.

There is an BugID for this RFE at [bugs.opensolaris.org](#). Hmm i hope this RFE will find it's way to Opensolaris or Solaris soon. Think about a Thumper as a Backup2Disk device with automatically dedups all data coming to it's disks. Or about an fileserver: Think about a combination of the in-kernel CIFS and the dedup functionality to save the storage needed for all this joke presentations.

(found via Robert Milkowski)

Posted by Joerg Moellenkamp at 07:35

Is there a way to make sure that a checksum cannot accidently be the same for different 2 files?
Or is the theory behind this "we just use the strongest algorithm we have - this will prevent us from that case"?
Anonymous on Apr 2 2008, 08:48

The only sure way would be a bit-wise comparison ... but this would be ineffective. Dedup (as far as i know all dedup products) is based on checksums. For the paranoid, two different checksum algorithms could be used.
Anonymous on Apr 2 2008, 09:44

You take the checksums only to identify possible blocks, which is the hard part. If you have the candidates bit-wise comparison is ok.
Anonymous on Apr 2 2008, 11:41

Yeap, this way you can save much of the performance. But it would be interesting to see the difference between bitwise compare and using multiple cryptographically strong hash algorithms ...
Anonymous on Apr 2 2008, 11:57

Here's an idea I'd love to see. Instead of making snapshots of an entire filesystem I'd love to see FILE based snapshots as a way to implement BACKUP VERSIONING. The way it would work would be you could set a flag on the filesystem or individual files so that EVERY TIME a file is opened, written to, then closed, a NEW copy of the file is created. I had versioning on an old VMS system and I miss it. It would keep the previous 'n' copies of a file as "filename;version".
Anonymous on Apr 3 2008, 17:24

This is what NetApps sis does, afaik.
Anonymous on Apr 4 2008, 17:44

Hmm, yeah ... sounds similar ... but A-SIS uses an own fingerprint database, the idea of Eric is the usage of the existing checksums.
Anonymous on Apr 4 2008, 18:13