ZFS and napp-it

# Data risk analysis

published: 2023-Apr-07 (c) napp-it.org

## A Data risk analysis
## Main risks for a dataloss listed by relevance.

**1. OMG, something happened to my data**
Human errors, Ransomware or sabotage by a former employee

- Last friday, i deleted a file. I need it now
- 6 weeks ago i was infected by Ransomware tat already encrypted some data
- 6 months ago data was modified by a former employee

**- Occurance: very often**
**- methods against:**
read only data versioning with at least a daily version for current week,
a weekly version for last month and a monthly version for last year

**Howto protect agaist:** Best solution is to use ZFS snaps, ou can hold thousands of readonly snaps without problem. They are created without delay and space consumption is only amount of modified datablocks fo a former snap. Only Unix root and not a Windows admin user can destroy ZFS snaps and not remote but only locally on server.

- **Restore**: simple, connect a ZFS filesystem via SMB und use Windows „previous versions" to restore single files or folders. On Solaris SMB „ZFS Snaps=previous versions" is zero config, with SAMBA you must care about snap folder settings.

or use ZFS rollback to set back a whole filesystem in time.

**Alternative**: use tape backups (hundreds of). A restore is very time consuming and can be quite complicated with differential backups.

**2. OMG, a diaster has happened**
**not the daily problems, a real disaster**

- A fire destroyed your server(s)
- A thieve has stolen your server(s)

**Occurance**: maybe never, but be prepared or anything can be lost
**methods against**: Create ongoing external daily disaster backups
Use disks or tapes or ZFS replicate to an external site or a removeable pool that you unplug/move after backup.

- Restore: simple but very time consuming. A restore of a large pool from backup can last days and without a current disaster backup data state is not recent.
As ZFS is a Unix filesystem, Windows AD SMB ACL permissions are not restored automatically ex on SAMBA but requires a correct mapping Unix uid -> Windows SID. The Solaris kernelbased server is not affected by mapping problems as it uses Windows SID directly as extended ZFS attributes.

If a disaster restore is not easy and straight forward with a simple copy method, test it prior an emergency case

**3. OMG, i missed a hardware failure for too lon**g
Servers are not „set and forget"

- a disk failed, then the next, then the pool
- a fan or climatisation failed and due overtemperature disks are damaged, data is corrupted

occurance: maybe once every few years
methods against: Monitor hardware and use mail alerts or you have a disaster case (see 2.)

**4. OMG, I cannot trust my data or backups**
suddently you discover a corrupted file or image with black areas, text errors or problems to open applications or files. Can you trust then any of your data?

occurance: prior ZFS sometimes, never with ZFS as ZFS protects data and metadata with checksums and repairs during read on the fly from Raid redundancy or on a regular base ex once every few months with a pool scrub.

**5. Ok, I cared about OMG problems with ZFS, many snaps and an external daily or weekly disaster backup Anything left to consider?**

There are indeed some remaining smaller problems even with ZFS

**- Server crashes during write:**

In this case the affected file is lost. ZFS can not protect you. With small files there is a minimal chance that data already completely in the rambased write cache. With sync enabled the file is written on next reboot.

Only the writing application can protect whole f iles against such incidents ex Word with temp files.
ZFS can protect the filesystem and commited writes.

**- incomplete atomic writes. Atomic writes are minimal dependent write operations that must be done completly.**
https://www.raid-recovery-guide.com/raid5-write-hole.aspx

An example is when a system crashes after data is written to storage and prior the needed update of metadata or when a database writes dependent transactions ex move money from one account to another with the result of dataloss or a corrupted filesystem.

In a Raid ex a mirror all data is written sequentially disk by disk. A sudden crash results in a currupted Raid/ mirror.

ZFS itself is not affected due Copy on Write where atomic writes are done completey or discarded at all so a currupted filesystem or Raid cannot occure by filesystem design. If you additionally want that any commited write is on save storage, you must enable sync.

If you have VMs with non Copy on Write filesystems, ZFS can guarantee for iteslf but not for guest operating systems. A crash during write can corrupt a VM. Activating sync write on ZFS can guarentee atomic writes and filesystems on VMs..

**-RAM errors,**
**Google for „ram error occurrence" about the risk**

All data is processed in RAM prior write. A RAM problem can corrupt these data that is then written to disk.
Even ZFS checksums cannot help as it can happen that you have bad data with proper checksums.

The risk of RAM errors is a small statistical risk that increases with RAM. A modern 32GB server has 64x the risk than a 512MB server with same quality of RAM. Only single errors are a problem. Bad RAM with many errors mostly results in an OS crash/ kernel panic or „too many errors" on reads with a disk or pool offline on ZFS. The „myth" of a ZFS scrub to death where ZFS wrongly repairs good data is a myth.

Anyway, if you care about data on a filer, always use ECC.
Even without ECC ZFS offers more security than older filesystems without ECC.

**-Silent data errors/ bit rot, Google „Bit Rot"**

This affects mostly long term storage with a statistical amount of data corruptions by chance over time. Some but not all can be repaired by the disk itself. ZFS can detect and repair all bitrot problems during read or a scrub of all data.
On long term storage, start regular scrubs ex once every few months to validate a pool prior problems become serious.

**- Insufficient redundancy.**

ZFS holds metadata twice and can hold data twice with a copies=2 setting even on single disks.
A ZFS raid offers a redundancy that counts in allowed disk failures until a pool is lost

As this is also a statistical problem, a rule of thumb is:
Best is when you allow any two disks to fail. This is the case with a raid-Z2 or 3way mirror. With more than say 10 disks per vdev, consider Z3. Slog or L2Arc do not need redundancy due a fallback to pool without dataloss. Only in case of an Slog failure combined with a system crash you see a loss of data in ramcache beside a performance degration.
If you use special vdevs, always use mirrors as a vdev lost means pool lost.

**- SSD without powerloss protection**

A crash during write on such SSDs can result in a dataloss or corrupted filesystem. If the SSD is part of a ZFS raid, problems can be repaired based on checksum errors, If the SSD is not in a Raid ex an Slog, SSD powerloss protection is mandatory.