iXsystems Technical Training

Erik Deridder
Sr. Systems Engineer

For iXsystems & Partner Use Only

TrueNAS CORE

OpenZFS

TrueNAS ENTERPRISE

iXsystems

TrueNAS
OPEN STORAGE

The Industry’s #1 Open Storage
>1 Million Deployments and 4-10 Exabytes
>250K User Community and #1 ZFS Distribution

Enterprise Storage with Open Source Economics
Enterprise features, security, integrations, & support
Manage data growth with cost-effective scalability
Choose software edition & hardware that fits the workload

Hybrid & All-Flash | Scale Up & Scale Out
Unified File, Block, Object, and App Storage
Industry-leading data management & CoW data integrity

EDITIONS

TrueNAS CORE
Free Single Node edition

TrueNAS ENTERPRISE
High Availability edition with Support

TrueNAS SCALE
Hyper-converged edition - *in development*

SYSTEMS

Minis | X-Series | M-Series

TrueNAS Product Family

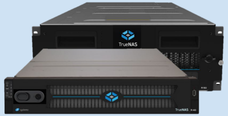
TrueCommand
Single Pane
Management

docker Cloud

TrueNAS Mini
No HA Support

Up to 130 TB

TrueNAS R-Series
No HA Support



Up to 4 PB

TrueNAS X-Series
HA Options Available

Up to 1 PB

TrueNAS M-Series
HA Options Available

Up to 22 PB

Community Support

12x5 Enterprise Support Available

24x7 Enterprise Support Available

File | Block | Object

Service Choices

NL-SAS | RI-SSD | SAS SSD | NVMe

Storage Media Choices

iXsystems

TrueNAS Product Family - Detailed Hardware Overview

TrueNAS Mini
No HA Support

Mini E:
- 4 bays
- 8 to 16GB DDR4
- Dual-core Intel Atom

Mini X:
- 5+2 bays
- 16 to 32GB DDR4
- Quad-core Intel Atom

Mini X+:
- 5+2 bays
- 32 to 64GB DDR4
- 8-core Intel Atom

Mini XL+:
- 8+1 bays
- 32 to 64GB DDR4
- 8-core Intel Atom

TrueNAS R-Series
No HA Support

R10:
- 1U 16x 2.5" 7mm bays
- 32 to 96GB DDR4
- 6 to 10 core Xeon SP

R20:
- 2U 12x 3.5" bays + 2x Exp. Shelf
- 32 to 192GB DDR4
- 6 to 16 core Xeon SP

R40:
- 2U 48x 2.5" 7mm bays + 2x Exp. Shelf
- 32 to 192GB DDR4
- 6 to 16 core Xeon SP

R50:
- 4U 48x 3.5", 3x NVMe bays + 2x Exp.
- 32 to 192GB DDR4
- 6 to 16 core Xeon SP

TrueNAS X-Series
HA Options Available

(All 2U 12x 3.5" bay)

X10:
- 2U 12x bays + 1x ES24
- 32 DDR4
- 6 core Xeon D

X20:
- 2U 12x bays + 1x ES60
- 64GB DDR4
- 6 core Xeon D

TrueNAS M-Series
HA Options Available

(All 4U 24x 3.5" bay)

M30:
- 64GB DDR4
- 8 core/8 thread Xeon SP

M40:
- 2x NVMe bays + 2x Exp. Shelf
- 128 to 192GB DDR4
- 10 core Xeon SP
- 1x 16GB NVDIMM

M50:
- 4x NVMe bays + 4x to 8x Exp. Shelf
- 256 to 384GB DDR4
- 2x 10 core Xeon SP
- 1x 16GB NVDIMM

M60:
- 4x NVMe bays + 8x to 12x Exp. Shelf
- 768GB DDR4
- 2x 16 core Xeon SP
- 2x 32GB NVDIMM

Expansion Shelves:
- 2U 12x 3.5" bay (ES12)
- 2U 24x 2.5" bay (ES24F)
- 4U 24x 3.5" bay (ES24)
- 4U 60x 3.5" bay (ES60)
- 4U 102x 3.5" bay (ES102)

(ES102 on R50 and M60 only, requires 1.2M deep rack)

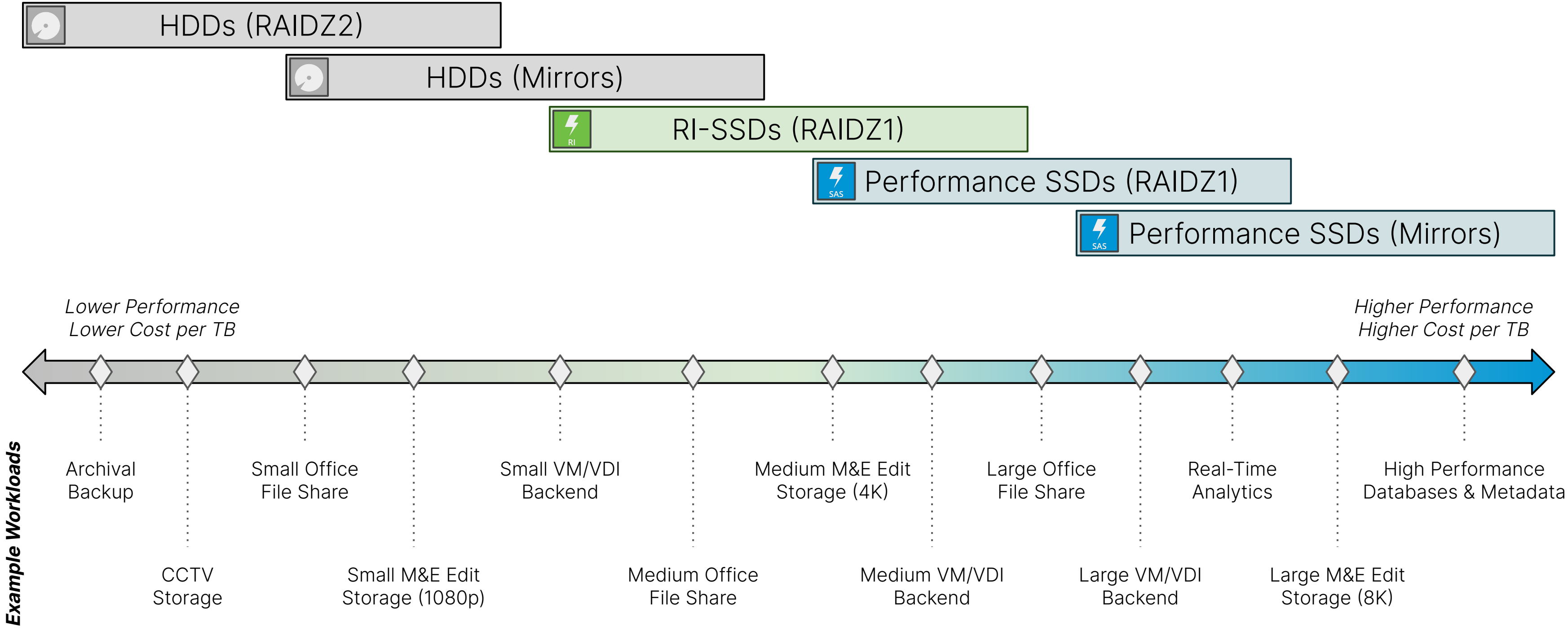
Network Options:
- 1GbE RJ45
- 10GbE RJ45
- 10GbE SFP+
- 40GbE QSFP+
- 25GbE SFP28
- 100GbE QSFP28
- 8 Gb/s FC
- 16 Gb/s FC
- 32 Gb/s FC

(Not all network options available on all platforms)

Encryption Support:
- Native OpenZFS Encryption
- Self-encrypting drives (SEDs)
- FIPS 140-2 SEDs

TrueNAS Media Options

Choose the media that suits your workload



Network Application Services

File
- ✓ NFS v3/4, SMBv1/2/3
- ✓ AFP, FTP, WebDAV, rsync

Block
- ✓ iSCSI, iSNS, FC, VAAI, Cinder, vCenter
- ✓ Certs: vSphere, Citrix, Veeam

Object
- ✓ S3 Host, Scale-out
- ✓ Cloud sync/backup, Credentials

Applications
- ✓ Plugins: Asigra, NextCloud, Iconik, Gitlab, Zoneminder, and more
- ✓ HA FreeBSD Jails/Plugins

Security
- ✓ Self-Encrypted Drives (SEDs)
- ✓ FIPS 140-2 Level 2 option
- ✓ Software disk encryption
- ✓ Encrypted replication, SSH
- ✓ Active Directory, LDAP
- ✓ Local users and groups
- ✓ Kerberos, ACLs, NIS

OpenZFS Services

Data Management
- ✓ Unlimited Snapshots & Restores
- ✓ Zero cost Clones
- ✓ Auto-restart ZFS Replication

Data Protection
- ✓ Copy-on-Write, 2-Copy Metadata
- ✓ RAID-Z1/Z2/Z3, Mirrors
- ✓ Fast Resilvering & Replication
- ✓ Data Integrity CRC, Scrubbing

Data Reduction
- ✓ Thin Provisioning
- ✓ In-line Compression
- ✓ Snapshot Clones

Data Acceleration
- ✓ All-Flash, Multiple Pools
- ✓ ARC, L2ARC Read Cache
- ✓ Write Cache - SAS or NVDIMM

TrueCommand
- ✓ Single pane of glass
- ✓ 24x7 Team operation
- ✓ RBAC, Audit, Single Sign-On (SSO)
- ✓ Alerting, Reporting, Analytics

Platform Services

Networking
- ✓ IPv4: 1- 100GbE, DHCP
- ✓ LAGG, VLANs, Firewall
- ✓ Fibre Channel (8-32Gb)

Administration
- ✓ Web UI, Wizards, SNMP, Syslog
- ✓ REST API, WebSockets API
- ✓ Alerting, Email, Support
- ✓ Tasks, Cron Jobs, Scripts, Reports
- ✓ In-Service Updates, Autotune

Operating System
- ✓ FreeBSD, Boot mgmt, SSH
- ✓ Iocage jails
- ✓ System logging, NTP

Hardware Management
- ✓ IPMI Remote Management
- ✓ SAS JBODs, Global hot spares
- ✓ SMART Drive Management

High Availability
- ✓ Dual Controllers, SAS, NVMe
- ✓ NVDIMM, Proactive Support
- ✓ Enclosure & Failover Management

The Z File System and OpenZFS

iXsystems Technical Training

iXsystems

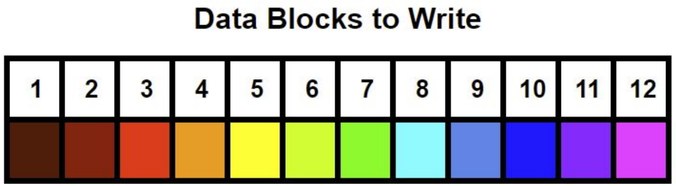
RAID, ZFS Pools, and Virtual Devices

- **RAID — R**edundant **A**rray of **I**ndependent **D**isks, a data storage technology to group hard drives for increased capacity and fault tolerance. Can be done in software or with a hardware controller.
- **OpenZFS —** Open source implementation of ZFS used in TrueNAS and FreeNAS
- **Virtual Devices (vdevs) —** Grouping of one or more hard disks. Vdevs are typically configured to group the disks using one of the following methods:
  - **Stripe:** Data is simply striped across all disks; no redundancy (like RAID 0)
  - **Mirror:** All disks in vdev get a copy of the data (like RAID 1)
  - **RAIDZ1:** Any number of data disks plus **one** parity disk (like RAID 5)
  - **RAIDZ2:** Any number of data disks plus **two** parity disks (like RAID 6)
  - **RAIDZ3:** Any number of data disks plus **three** parity disks
- **ZFS Pool —** Sometimes called a "zpool" or simply a "pool", this consists of one or more vdevs are striped together (like RAID 0). For example, multiple mirrored vdevs in a pool behave like RAID 10 (or RAID 1+0). Multiple RAIDZ2 vdevs in a pool behave like RAID 60 (or RAID 6+0).
- Vdev and pool configurations will have a major effect on overall storage **capacity**, **reliability**, and **performance**. Each configuration, its drawbacks, and its advantages will be discussed in the following slides.

If one or more vdevs in a pool fails, the data in the pool will be lost!

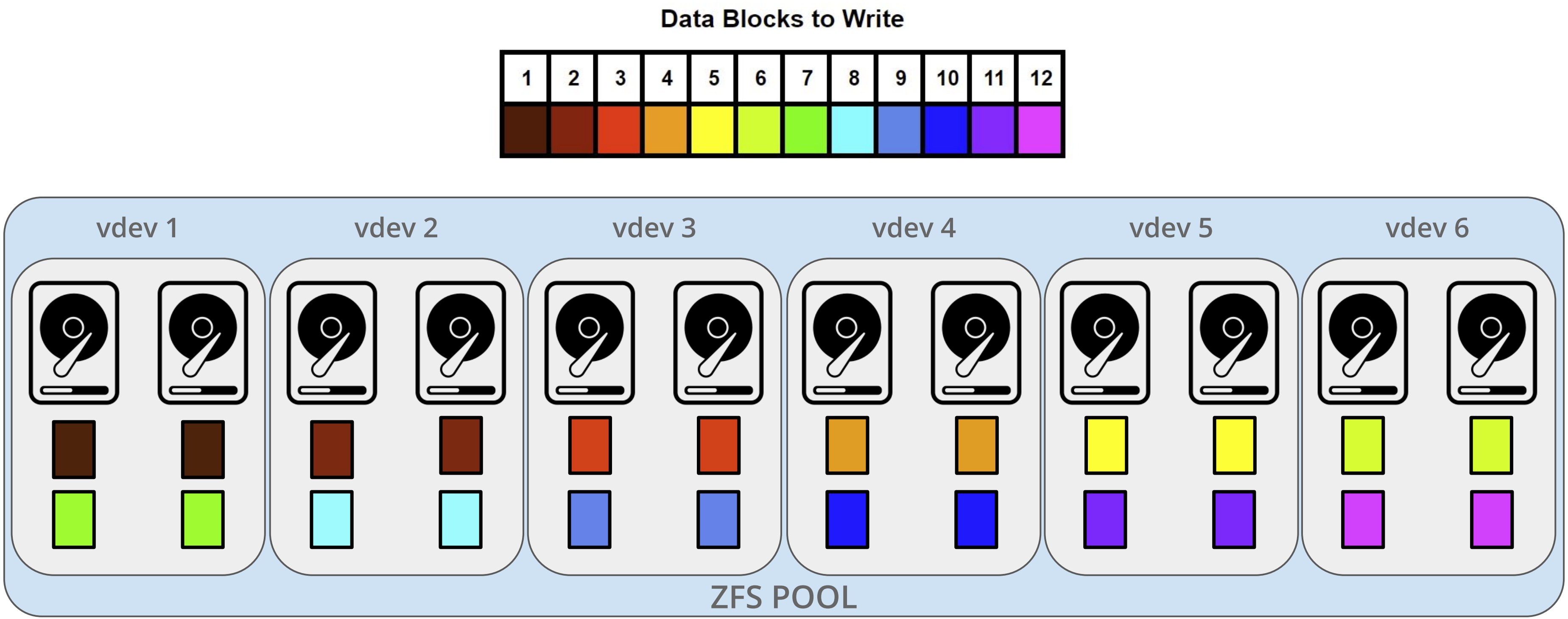
Vdev Reliability Examples

- In the following slides, we'll go through examples of how each vdev type spreads its data across its drives
- When data is written to a ZFS pool, it is broken up into smaller chunks called "blocks". The size of those blocks can actually vary depending on the file size, but we won't worry about that. Just know that ZFS breaks up data into "blocks".
- We'll write data to the pools represented by a series of rainbow blocks, with one color representing on block of a file we're writing:



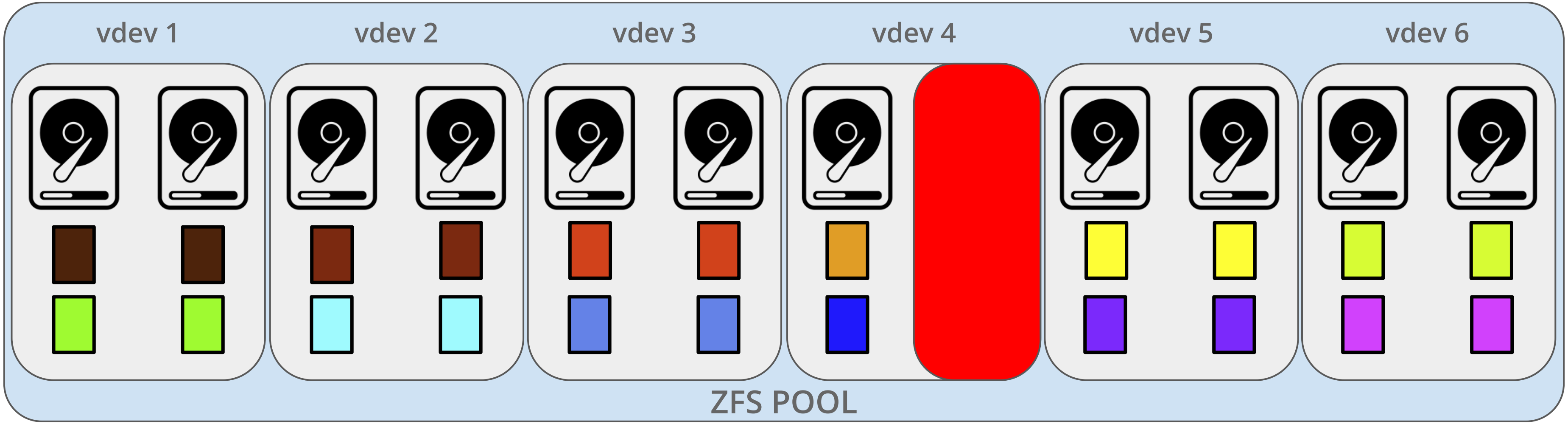
- We'll then see what happens to this data as disks fail. Our objective will always be to reconstruct the rainbow in spite of disk failures.
- The RAIDZ examples will do something a little different, but we'll discuss that when we get there...

Mirrored Vdev Example



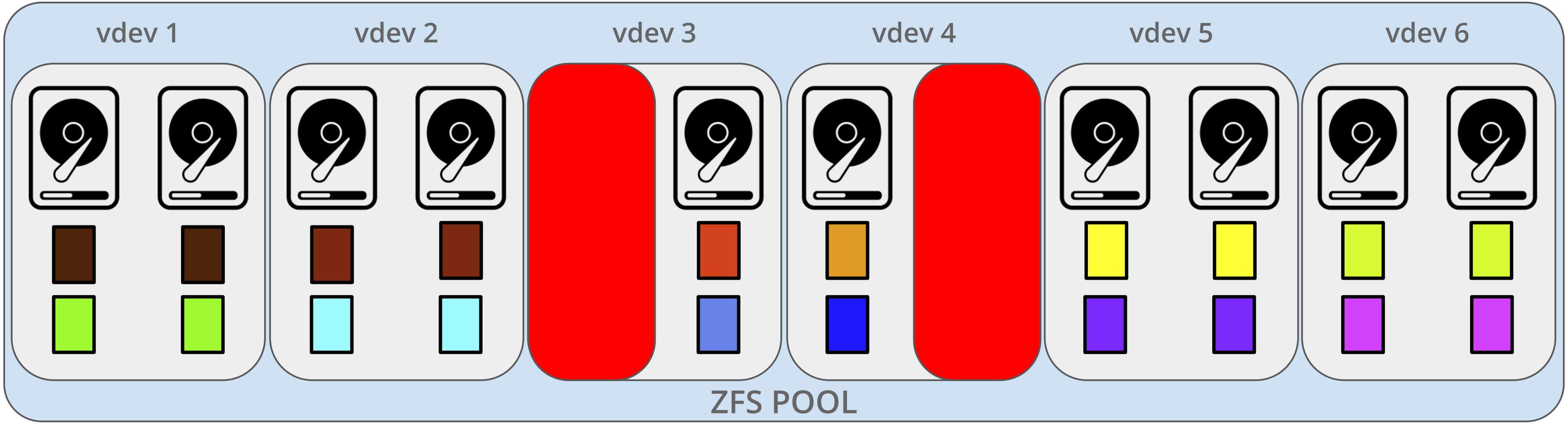
Mirrored Vdev Example

- Data is striped across all vdevs just like it was in the previous example, except each vdev now has two disks.
- In a mirrored vdev, each disk in the vdev gets one full copy of all the data written to the vdev, adding redundancy!
- Can lose all but one disk per vdev and not suffer data loss.



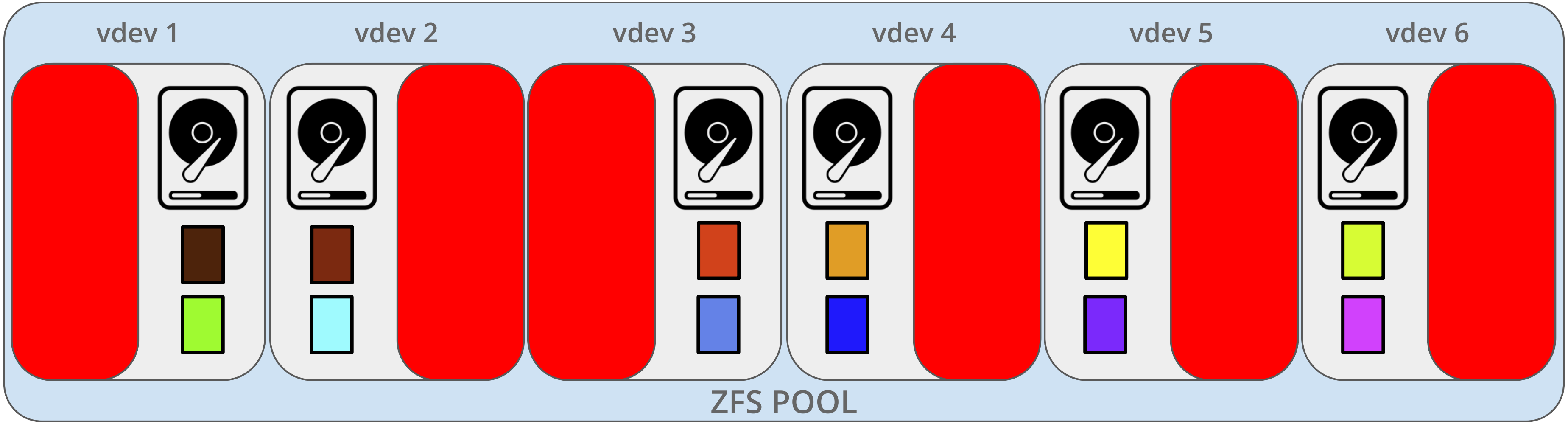
Mirrored Vdev Example

- Data is striped across all vdevs just like it was in the previous example, except each vdev now has two disks.
- In a mirrored vdev, each disk in the vdev gets one full copy of all the data written to the vdev, adding redundancy!
- Can lose all but one disk per vdev and not suffer data loss.
- Can lose multiple disks per pool, as long as no one vdev is completely lost (i.e., all drives in that vdev lost).



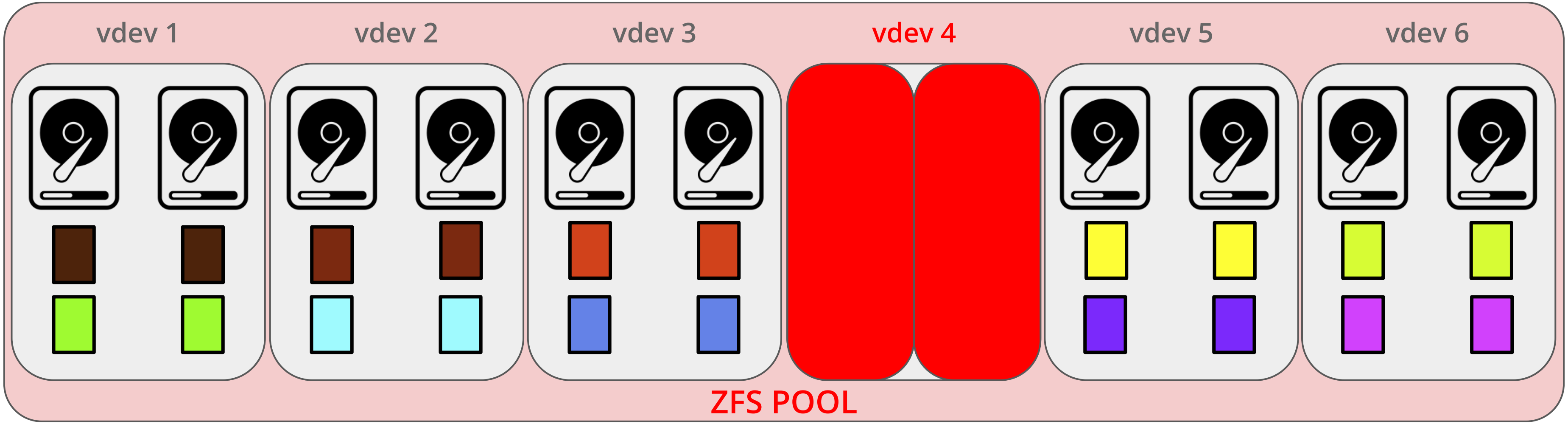
Mirrored Vdev Example

- Data is striped across all vdevs just like it was in the previous example, except each vdev now has two disks.
- In a mirrored vdev, each disk in the vdev gets one full copy of all the data written to the vdev, adding redundancy!
- Can lose all but one disk per vdev and not suffer data loss.
- Can lose multiple disks per pool, as long as no one vdev is completely lost (i.e., all drives in that vdev lost).



Mirrored Vdev Example

- Data is striped across all vdevs just like it was in the previous example, except each vdev now has two disks.
- In a mirrored vdev, each disk in the vdev gets one full copy of all the data written to the vdev, adding redundancy!
- Can lose all but one disk per vdev and not suffer data loss.
- Can lose multiple disks per pool, as long as no one vdev is completely lost (i.e., all drives in that vdev lost).
- All pool data is lost if any one mirror fails (i.e., all disks in the mirror fail).



iXsystems

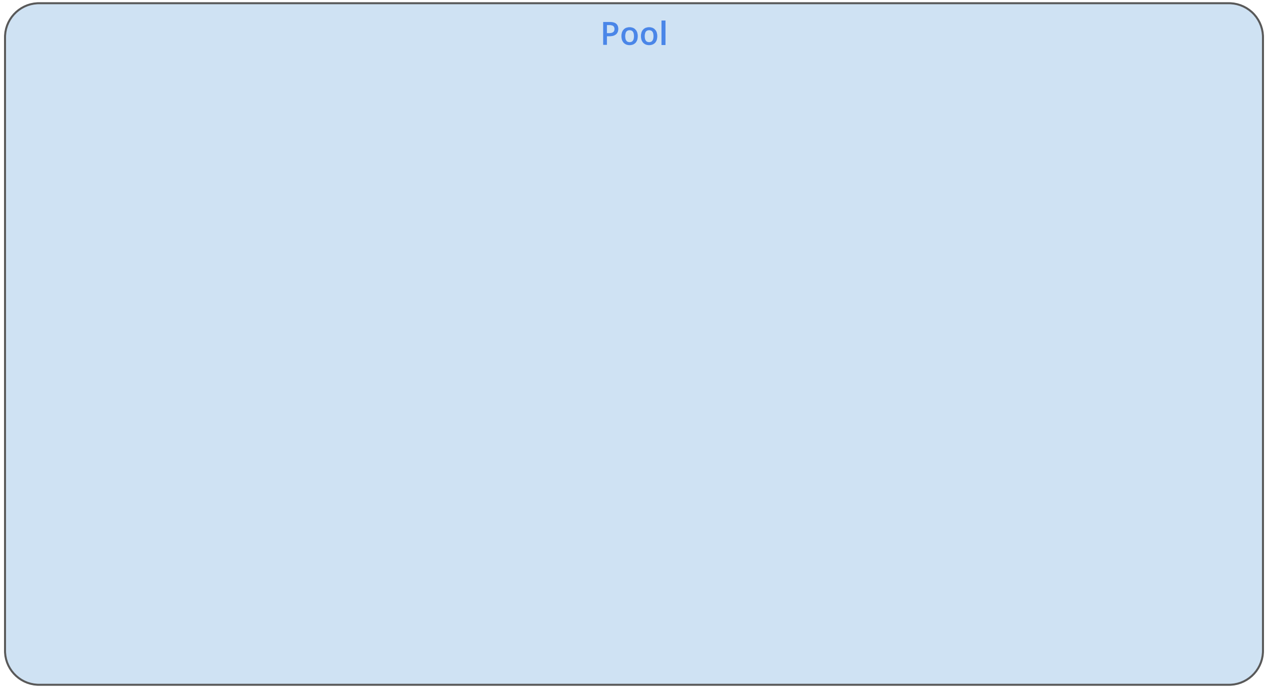
ZFS Datasets & ZVols

- Now that we have our pool, we can store data on it. We have a couple different types of "containers" in which we can store the data: **datasets** and **zvols**
- **Datasets** contain file systems with folders and files that you can browse through. This is what you would typically think of when you store data on a hard drive.
    - A dataset acts kind of like a folder that contains all your stuff, but you can set more detailed attributes on it, like capacity limits, compression type, dedupe, sync settings, etc.
    - You can nest datasets (i.e., a dataset inside another dataset) just like you can nest normal folders. The "child" dataset can have different settings than the parent (e.g., no compression).
    - File-sharing protocols like SMB, AFP, and NFS will work out of datasets.
- **Zvols** are simply a chunk of raw disk space. ZFS just keeps track of the 1s and 0s on it.
    - Zvols are created inside a dataset but don't appear as a file or folder in that dataset; they just take up a chunk of disk space.
    - The ZFS system doesn't know what files are on the zvol, it's just tracking the 1s and 0s
    - Block protocols like iSCSI and FC use zvols; client will create the file system on the raw space.

iXsystems™

ZFS Pools, Vdevs, Datasets, & Zvols

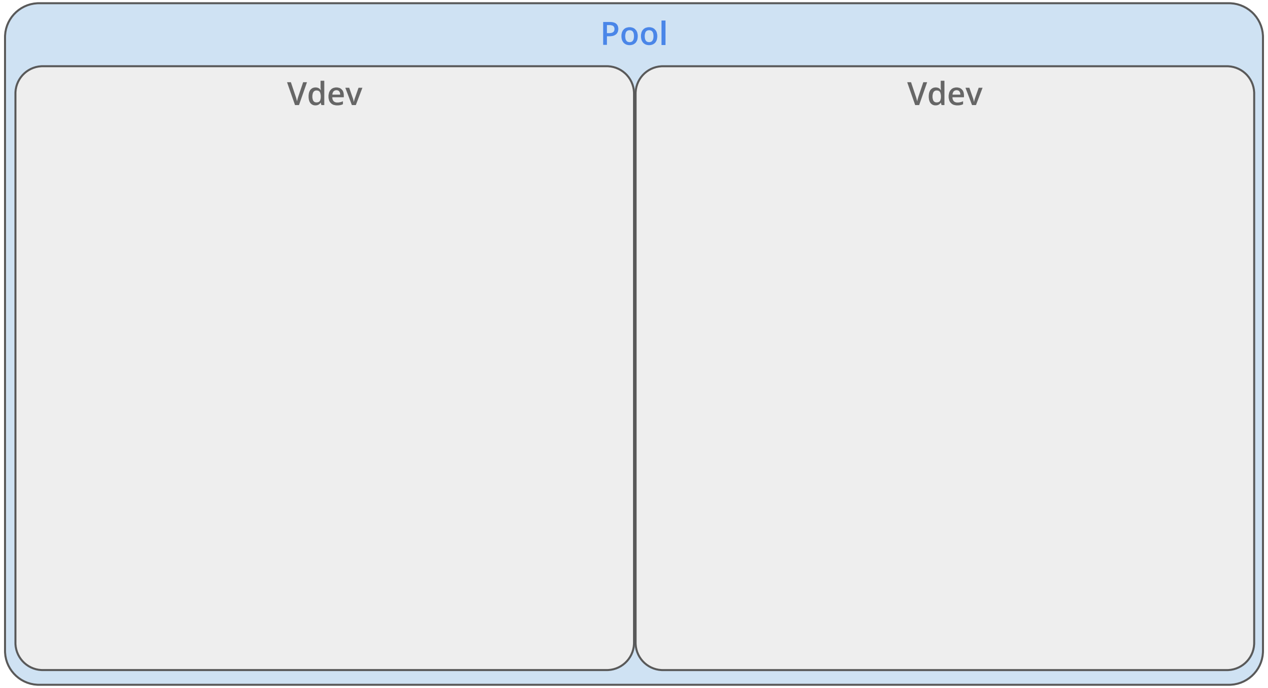
- Let’s review all the building blocks we just covered and look at how they all fit together in a ZFS system:
- At the top of the hierarchy, we have our pool



ZFS Pools, Vdevs, Datasets, & Zvols

- Let's review all the building blocks we just covered and look at how they all fit together in a ZFS system:

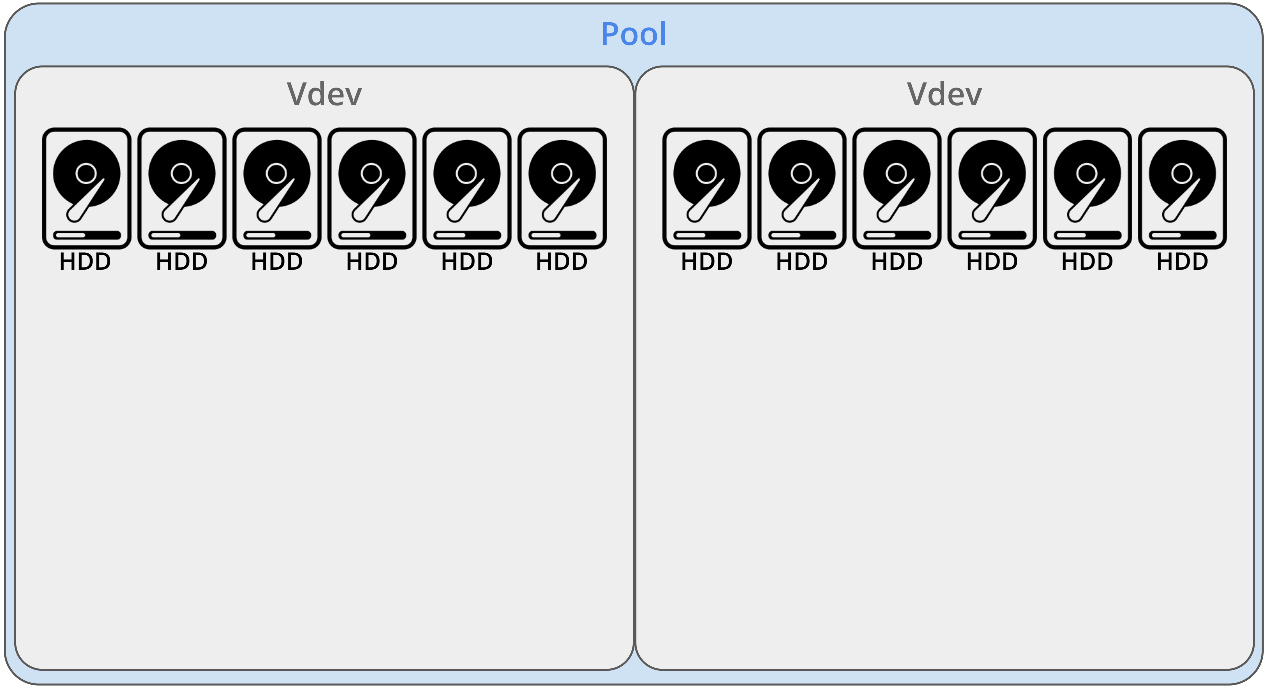
- At the top of the hierarchy, we have our pool
- The pool is made up of one or more vdevs



ZFS Pools, Vdevs, Datasets, & Zvols

- Let’s review all the building blocks we just covered and look at how they all fit together in a ZFS system:

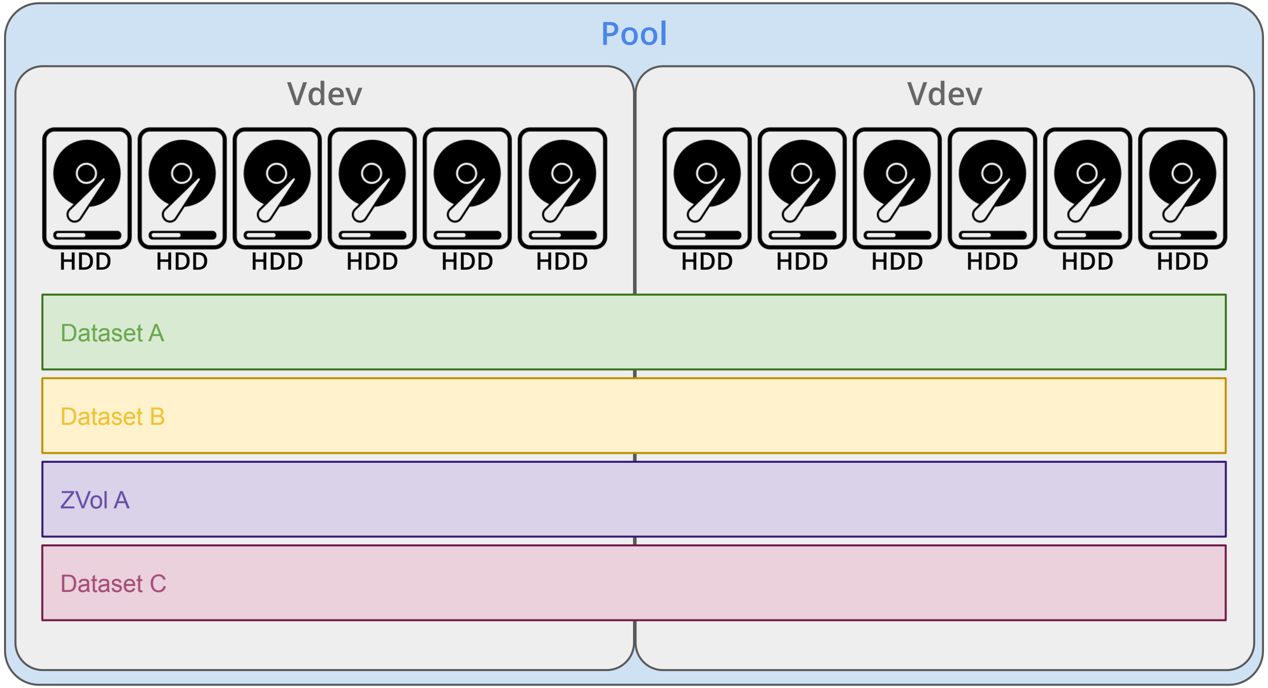
- At the top of the hierarchy, we have our pool
- The pool is made up of one or more vdevs
- Each vdev consists of one or more hard drives grouped together in some configuration



ZFS Pools, Vdevs, Datasets, & Zvols

- Let's review all the building blocks we just covered and look at how they all fit together in a ZFS system:

- At the top of the hierarchy, we have our pool
- The pool is made up of one or more vdevs
- Each vdev consists of one or more hard drives grouped together in some configuration
- Datasets & Zvols are striped across all the Vdevs

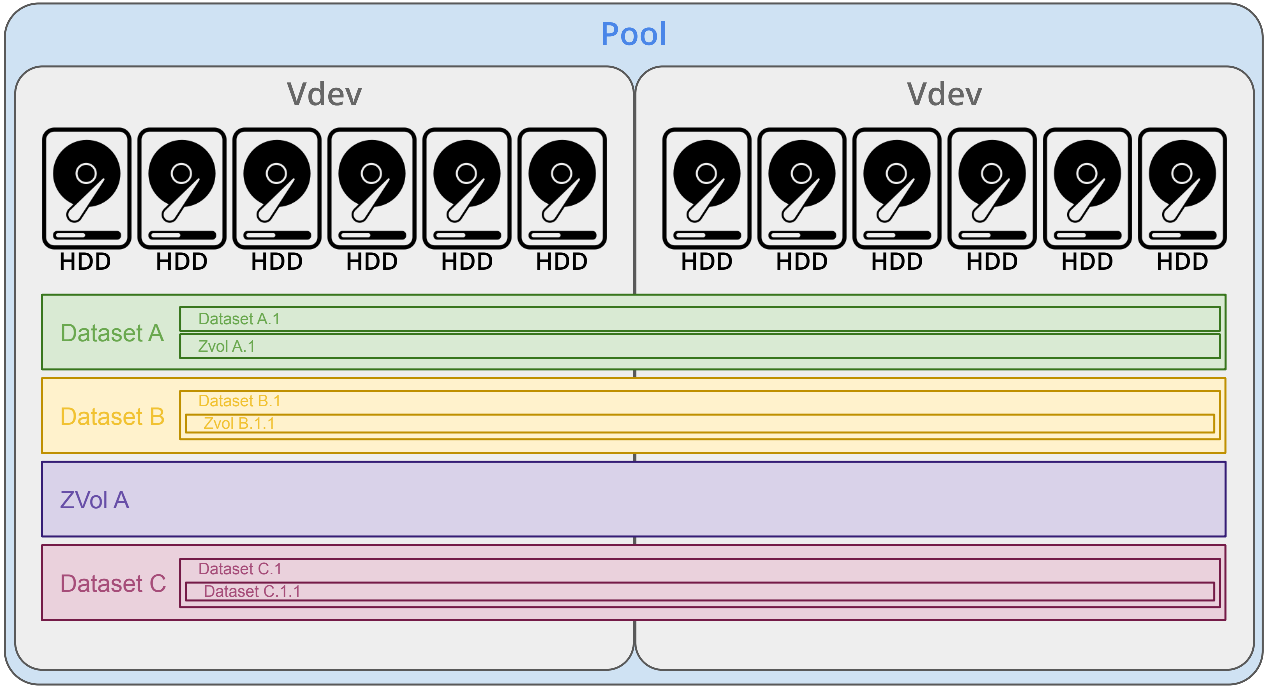


iXsystems

ZFS Pools, Vdevs, Datasets, & Zvols

- Let's review all the building blocks we just covered and look at how they all fit together in a ZFS system:

- At the top of the hierarchy, we have our pool
- The pool is made up of one or more vdevs
- Each vdev consists of one or more hard drives grouped together in some configuration
- Datasets & Zvols are striped across all the Vdevs
- Datasets can be repeatedly nested inside of other datasets
- Zvols can be nested in datasets, but nothing can be nested in a Zvol
- All of these children and grandchildren are also striped across all the drives



Modifying Pools, Vdevs, & Datasets

ZFS allows you to perform some modifications on pools, vdevs, datasets, and zvols after they have been created. Here's what you're allowed to and not allowed to do:

- Modifications **Allowed** to Pools:
  - Add more vdevs to the pool
  - Add SLOG to pool
  - Remove SLOG from pool
  - Add L2ARC to pool
  - Remove L2ARC from pool
  - Destroy the pool

- Modifications **Not Allowed** to Pools:
  - Remove vdevs from the pool
  - Enable encryption
  - Disable encryption

- Modifications **Allowed** to Vdevs:
  - Add a drive to mirrored vdev (e.g., go from 2-way to 3-way mirror)
  - Remove a drive from a mirrored vdev (e.g., go from 3-way to 2-way mirror)
  - Break a mirrored vdev (e.g., go from 2-way to striped)
  - Destroy the Vdev (by destroying the pool)
  - Replace individual disks in the vdev

- Modifications **Not Allowed** to Vdevs:
  - Change the configuration of a vdev (e.g., go from Z2 to Z3)
  - Add drives to a Z1, Z2, or Z3
  - Remove drives from a Z1, Z2, or Z3

- Modifications **Allowed** to Datasets/Zvols:
  - Add dataset under root or under another dataset
  - Add zvol under another dataset
  - Change **most** dataset and zvol properties (e.g., compression setting, recordsize value)*****
  - Destroy dataset or zvol

- Modifications **Not Allowed** to Datasets/Zvols:
  - Change dataset or zvol name
  - Change dataset case sensitivity
  - Change zvol block size and sparse volume settings
  - Add zvol at root pool level
  - Add dataset under a zvol

*** Usually not retroactive; change will only apply for data newly-written to dataset or zvol**

iXsystems

Thank you