OpenZFS and Linux
Who is this guy?

Nikolai Lusan

Email: nikolai @_@ lusan.id.au

IRC: Maliuta
    on Freenode and OFTC
OpenZFS
Now With Native Encryption!
CDDL and GPL are considered incompatible, most distributions will not build installers for containing ZFS support.
Filesystems aren’t they fun?

No.
No.
No.
They are not.

Why Not?

Data Loss
Bit Rot
Performance
Why ZFS?

- It’s cool
- Stable and established
- Robust
- Good performance - even better with tuning
- Scales up well
- Allows better usage of disk space
- More features than other file systems
Why ZFS?

- Designed with systems administrators in mind
- Changes approach to data storage
- Works well in bare metal and virtual environments
- Built in ability to share storage via almost any method available under Linux - eg. NFS, SMB, iSCSI ...
“Traditional” Filesystem Layout

Filesystems

Logical Volumes

LUKS 1 LUKS 2 LUKS 3 LUKS 4 LUKS N

Encryption

RAID 1 RAID 2 RAID 3 RAID 4 RAID N

RAID

Raw Disk

Disk 1 Disk 2 Disk 3 Disk 4 Disk 5 Disk 6 Disk 7 Disk 8 Disk N
How ZFS Architecture is Different

- Copy on Write (CoW)
- Abstracts storage from disks
- Has internal measures that replace traditional Linux file system access
- Uses pools of virtual devices (VDEV’s) which can be of different size and underlying implementation
- Data is stored in datasets, these are similar to LVM logical volumes but far more configurable
ZFS Approach

Datasets

Pool 1

VDEV 1
VDEV 2
VDEV 3
VDEV N

Pool 2

VDEV 1
VDEV 2
VDEV 3
VDEV N

Raw

Disk

Disk 1
Disk 2
Disk 3
Disk 4
Disk 5
Disk 6
Disk 7
Disk 8
Disk N
VDEVS

• VDEVS are “Virtual Devices”
• They can have different geometries
  - Single disk
  - Mirror of 2 or more disks
  - Multiple types of RAID
• VDEVS are pooled together to create usable storage space
• Writes are striped across VDEVS
• Losing a VDEV means losing data
Pools

• Made up of one or more vdevs
• Writes spread over vdevs
• Mountable filesystem in it’s own right
• Many pool level attributes are inherited by datasets
• Pools can be moved from one machine to another with minimal hassle
• When creating pools remember some settings are immutable
Datasets

- Created from ZFS pools
- Each has a set of tuneable attributes
  - Some attribute cannot be changed from inherited, or initial values
- Mountable in arbitrary locations
ZVOLs

- Block devices
- Multiple uses, including swap
- Arbitrary block size
- Not as performant as raw datasets
- Can be exposed to the OS in different ways
ARC/L2ARC/SLOG(ZIL)

- ARC is Adaptive Replacement Cache
- L2ARC is Layer 2 ARC - taken from RAM moved to disk
- SLOG or ZFS Intent Log (ZIL) is an intermediate journal of disk writes that are yet to happen. It allows for a write acknowledgment to be sent to applications/OS faster. ZIL also acts as a kind of journal preventing data loss between boots.
ZFS Tools

- zpool
- zfs
- zed
- zdb
Creating VDEVs

- Use disk/partition names that will remain constant
- Remember not all disks need to be of the same size
- Not all VDEVs need to be of the same type
- There are 3 vdev types
  - Single disk/partition
  - Mirror (with no limit on the number of devices)
  - RAIDZ (with the option for up to triple parity)
Snapshots

- Provide a glimpse of the dataset at the time taken
- Can be used to roll back a dataset to the point in time the snapshot was created
- Mountable
- They do take up space
- The space used is only a delta from the most recent snapshot
- Not automatically deleted, so they need to be managed
- There are existing tools to automatically manage snapshots
  - zsnapd
  - zfs-auto-snapshot
- Can be enabled/disabled per dataset
Snapshots for Offsite Backup

- The “zfs” tool provides a send function and a receive function allowing snapshots to be sent between pools.

- The pools do not need to be on the same machine, the receiver can even be a dataset under another pool (e.g. send from `<pool>/dataset@<snapshot>` to `<pool_2>/dataset/dataset_2`).

- Most common transport is via ssh, but any tool that lets you send and receive data can be used (mbuffer is another common tool).
Tuneables

- There are almost 230 tuneable parameters for the kernel module alone.
- There are over 75 tuneable parameters for each dataset, more when you are dealing with enabling non-standard or new features.
Compression and Deduplication

- Native filesystem level compression
  - lz4
  - lzjb
  - gzip
  - zle
- Deduplication is RAM intensive (1GB of RAM for every 1TB of deduplicated data space)
- Both can help you squeeze more storage out of your disk
Optimisation for all ZFS (Kernel)

- Tune the ARC size to fit your needs
- Tune metaslab performance for spreading writes across vdevs
- Tune ARC/L2ARC performance
- Tune TRIM limits for SSD storage
Easy Tuning for Most Purposes

- Create pools using ashift=12
- Enable lz4 compression
- Set recordsize to 128k
- Disable atime, dev, exec, suid as needed (atime is a big saver)
- Set logbias to latency
- Set sync to “standard” or “disabled”
Optimisation for MySQL/MariaDB

- This is for innodb only MyISAM is left to people who know this RDBMS better
- recordsize=16k
- primarycache=metadata
- logbias=throughput
Optimisation for PostgreSQL

- Use separate datasets for data and WAL
- recordsize=8k
- primarycache=metadata
- logbias=throughput
Optimisation for running VM’s

- Controversy over using vdev’s versus qcow2 files
- Different approaches require different optimisation
- VDEV’s should be created with a recordsize that reflects the FS that will run on the VM, have logbias=throughput, and primarycache=metadata and volmode=full
- Using qcow2 files on dedicated datasets is the recommended way. The datasets should have a recordsize that matches the FS that will be used in the VM
Running ZFS in a [hosted] VM

- Use a single disk vdev
- Still use an SLOG device
- Worry more about file compression and RAM usage than underlying storage.
Resources

• **Manpages**
  - zpool
  - zfs
  - zdb
  - zpool-features
  - zfs-module-parameters
  - zfs-events

• **Online**
  - OpenZFS wiki [http://open-zfs.org/wiki/Main_Page](http://open-zfs.org/wiki/Main_Page)
  - ZFS on Linux FAQ [https://github.com/zfsonlinux/zfs/wiki/FAQ](https://github.com/zfsonlinux/zfs/wiki/FAQ)
The End ...