



Linux Clusters Institute: ZFS Hands On Exercise

Georgia Tech, August 15th – 18th 2017

J.D. Maloney | Storage Engineer
National Center for Supercomputing Applications (NCSA)
malone12@illinois.edu



Goal of Hands on Exercise

- Walk through ZFS install
- Build vdev_id.conf file
- Create zpools of different types
- Get familiar with zpool attributes
- Experiment with compression
- Set up snapshots/restore from them
- Set up zpool scrubs



ZFS Install

- Grab the zfs repo & install gpg key

```
[root@zfs-demo ~]# wget http://download.zfsonlinux.org/epel/zfs-release.el7_3.noarch.rpm
[root@zfs-demo ~]# gpg --quiet --with-fingerprint /etc/pki/rpm-gpg/RPM-GPG-KEY-zfsonlinux
gpg: new configuration file `/root/.gnupg/gpg.conf' created
```

- Install the repo

```
[root@zfs-demo ~]# rpm -ivh zfs-release.el7_3.noarch.rpm
Preparing...                               ##### [100%]
Updating / installing...
 1:zfs-release-1-4.el7_3.centos             ##### [100%]
[root@zfs-demo ~]# █
```

- Install zfs and kernel-devel

```
[root@zfs-demo ~]# yum install zfs kernel-devel
```

- Load the kernel module & Enable module on boot

```
[root@zfs-demo ~]# /sbin/modprobe zfs
[root@zfs-demo ~]# systemctl enable zfs-import-cache zfs-import-scan zfs-mount zfs-share zfs-zed zfs.target
```

- Check to make sure all is happy

```
[root@zfs-demo ~]# zpool status
no pools available
```

Creating vdev_id.conf file

- Find devices

```
[root@zfs-demo ~]# fdisk -l
```

```
Disk /dev/vda: 42.9 GB, 42949672960 bytes, 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x0000c4ba
```

```
   Device Boot      Start         End      Blocks   Id  System
/dev/vda1 *          2048     83875364   41936658+  83  Linux
```

```
Disk /dev/vdb: 5368 MB, 5368709120 bytes, 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disk /dev/vdc: 5368 MB, 5368709120 bytes, 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disk /dev/vdd: 5368 MB, 5368709120 bytes, 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disk /dev/vde: 5368 MB, 5368709120 bytes, 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Devices are /dev/vd* ; Non-VM environments usually are /dev/sd* or dm-* from multipath

Creating vdev_id.conf file

- Find devices

```
[root@zfs-demo ~]# for i in {b..e}; do echo vd${i} && udevadm info --query=property /dev/vd${i} | grep DEVLINKS; done
vdb
DEVLINKS=/dev/disk/by-id/virtio-cab28228-a255-4da4-9
vdc
DEVLINKS=/dev/disk/by-id/virtio-9d585a48-8667-4668-b
vdd
DEVLINKS=/dev/disk/by-id/virtio-ec112144-7b6e-41e0-a
vde
DEVLINKS=/dev/disk/by-id/virtio-e693dbfd-a98e-4d81-a
```

- Create vdev_id.conf File

```
multipath          no
topology           sas_direct
alias  slot_0      /dev/disk/by-path/virtio-pci-0000:00:06.0
alias  slot_1      /dev/disk/by-path/virtio-pci-0000:00:07.0
alias  slot_2      /dev/disk/by-path/virtio-pci-0000:00:08.0
alias  slot_3      /dev/disk/by-path/virtio-pci-0000:00:09.0
```

Creating zpools

- Create different kinds of zpools, an example below

```
[root@zfs-demo ~]# zpool create fs_0 mirror slot_0 slot_1 mirror slot_2 slot_3 -f
```

- Use the zpool destroy command between zpools

Check Out zpool properties

- Check out and play with properties
 - Some good ones: recordsize, readonly, snapdir, sync

```
[root@zfs-demo ~]# zfs get all fs_0
```

NAME	PROPERTY	VALUE	SOURCE
fs_0	type	filesystem	-
fs_0	creation	Mon Jun 26 10:25 2017	-
fs_0	used	56.5K	-
fs_0	available	9.63G	-
fs_0	referenced	19K	-
fs_0	compressratio	1.00x	-
fs_0	mounted	yes	-
fs_0	quota	none	default
fs_0	reservation	none	default
fs_0	recordsize	128K	default
fs_0	mountpoint	/fs_0	default

ZFS Compression

- Create a zpool (doesn't matter geometry)
- Turn on lz4 compression
- Rsync over the data again
- Check the used space with lz4 compression
- Delete data and try with other algorithms

ZFS Snapshots

- On an empty zpool take a snapshot
- Rsync in some data from sample dataset
- Take another snapshot of the zpool
- Delete a subset of data that you copied over
- Verify it's gone
- Rollback to the snapshot you took
- Verify data is back

ZFS Scrubs

- Have a zpool with data on it
- Create a new script file
- Drop in the scrub command for that zpool (full command path)
- Run the script manually
- Verify it is running
- Can be put in cron via cron method of choice

Wrap Up

- Further Exploration
 - Quotas
 - ZFS Send/Receive
- When done playing with ZFS
 - Destroy any zpools you created
 - Leave ZFS installed, we'll come back to it later

