EPJ Web of Conferences **214**, 04043 (2019) https://doi.org/10.1051/epjconf/201921404043
CHEP 2018

Using ZFS to manage Grid storage and improve middleware resilience

Robert Currie[1,*], *Teng* Li[1,**], and *Andrew* Washbrook[1,***]

[1]Edinburgh University, James Clerk Maxwell Building, Peter Guthrie Tait Road, Edinburgh, EH9 3FD

Abstract. ZFS is a powerful storage management technology combining filesystem, volume management and software RAID technology into a single solution. The WLCG Tier-2 computing at Edinburgh was an early adopter of ZFS on Linux, with this technology being used to manage all of our storage systems including servers with aging components. Our experiences of ZFS deployment have been shared with the Grid storage community which has led to additional sites adopting this technology.
ZFS is highly configurable therefore allowing systems to be tuned to give best performance under diverse workloads. This paper highlights our experiences in tuning our systems for best performance when combining ZFS with DPM storage management. This has resulted in reduced system load and better data throughput. This configuration also provides the high redundancy required for managing older storage servers. We also demonstrate how ZFS can be combined with Distributed Replicated Block Device (DRBD) technology to provide a performant and resilient hypervisor solution to host multiple production Grid services.

1 Introduction

In order to meet the requirements of running a WLCG Tier-2, resources hosted by a university have to provide a high level of reliability. To most efficiently provide this with the minimum amount of manpower it is best to build automation, failover and reliability into the sites infrastructure. With this in mind Edinburgh has chosen to make use of ZFS[1] as a backing storage technology.

Building on previous work[2] at Edinburgh, ZFS has been used as the main storage management solution at the Tier-2. This is now used to manage the various storage servers at the site to manage aging and newer storage technologies. As well as being a purely storage management solution, we have combined ZFS with DRBD to provide a network redundant storage system to store Virtual Machine(VM) images which host Tier-2 services.

2 ZFS best practices

ZFS is a storage management solution originally developed by Sun Microsystems in the early 2000's for the Solaris platform. This solution differs from most other storage systems as it

*e-mail: rcurrie@cern.ch
**e-mail: Teng.Li@ed.ac.uk
***e-mail: awashbro@ph.ed.ac.uk

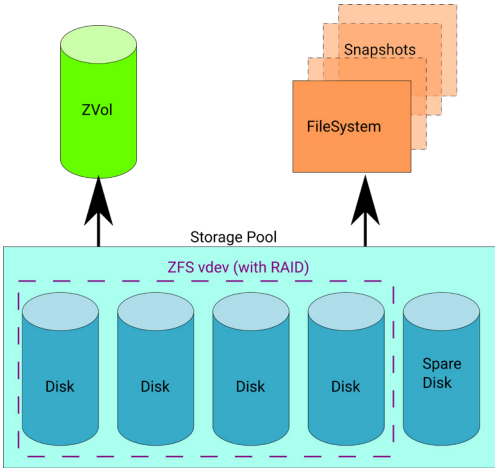

Figure 1. A basic diagram of some of the main components within the ZFS system. Storage disks are managed by vdevs which are used to construct pools of data. In turn these pools are exposed to the operating system as either filesystems or as virtual block devices known as ZVols. Here the pool of data has 1 redundant disk which is not in the main storage but the main vdev may be configured with varying levels of redundancy between RAIDz1, RAIDz2 or RAIDz3.

mixes volume management, software RAID and filesystems into the same piece of software. ZFS is particularly attractive due to it's design emphasis on data integrity and reliability.

ZFS makes storage available to Linux though mount points which provide a POSIX like interface to the storage managed by some combination of vdevs as in Figure 1. Redundancy within a ZFS filesystem is provided through the configuration of the vdev device. The most common configuration of this for larger storage configurations with many disks is "RAIDz3". This configuration provides 3 disks of redundancy for data stored within the vdev meaning that it can withstand the loss of 3 physical disks without data loss. Additional disks may also be added to provide greater redundancy and reduce the urgency of interventions to address disk loss.

Typically it is considered best practice to split disks across multiple vdevs and to construct a pool of storage from these. However, when deploying the Tier-2 in Edinburgh we deployed only 1 large vdev per storage server. This allowed us to maximise the amount of available storage and to reduce the complexity of maintenance of multiple server configurations.

3 Tuning ZFS for HEP workloads

When combining a large number of disks into a single vdev (Figure 1), it was found that using ZFS to administer storage at a Tier-2 benefited from further optimizations. Storage access patterns within the Edinburgh Tier-2 were found to be strongly non-uniform in time with the number and size of files varying dramatically. In order to cope with periods when storage servers were being taxed by many parallel file operations several ZFS parameters were tuned to improve performance with our workloads.

The main changes to the default ZFS configuration were to increase the number of parallel operations that can be performed on a single vdev as well as to reduce the load associated with removing large amounts of data from the storage. A summary of the parameters tuned

Parameter	default value	optimized value	Description
`zfs_vdev_(a)sync_read_min_active`	1	12	Minimum number of (a)synchronous reads which can happen in parallel on a vdev.
`zfs_vdev_(a)sync_write_min_active`	1	12	Minimum number of (a)synchronous writes which can happen in parallel on a vdev.
`zfs_vdev_(a)sync_read_max_active`	10	256	Maximum number of (a)synchronous reads which can happen in parallel on a vdev.
`zfs_vdev_(a)sync_write_max_active`	10	256	Maximum number of (a)synchronous reads which can happen in parallel on a vdev.
`zfs_top_maxinflight`	32	1,024	This is the maximum number of parallel operations to happen at once on a single vdev.
`zfs_delete_blocks`	20,480	512	Files larger than `zfs_delete_blocks` * block size are removed synchronously
`zfs_txg_timeout`	500,000	250,000	This parameter impacts how long async data should reside in RAM before ZFS attempts to flush it to disk.

Table 1. Table of the parameters which have been tuned to optimize ZFS for Tier-2 workloads at Edinburgh. This describes the default value of these parameters, the optimized values used in production and what these parameters control.

and their descriptions are given in Table 1. The impact of these optimizations was to reduce server load as well as to improve stability and performance of WLCG software on the storage servers.

4 Running Tier-2 resources on a Hypervisor(HV)

At Edinburgh virtualizing the Tier-2 resources was found to provide several advantages. The first of these was to reduce the total number of servers required to provision the Tier-2 resources providing compute/storage access. The second was that provisioning servers this way offered the most efficient and reliable way of deploying and maintaining services.

To build a HV designed to provide the maximum level of redundancy the 2 host systems were provisioned with 128GB of RAM and each had 2 physical drives for hosting VMs. In addition to being externally accessible over the network these machines were also configured with a second 10Gbps link dedicated to traffic associated with the storage backups.

In order to protect against hardware failure, the 2 HV systems have been provisioned with both local and network based redundancy. The key software components used to deploy this are ZFS and DRBD[3], and the design of this storage solution is shown in Figure 2. Each server has 2 underlying physical disks (sda and sdb) which are configured within a ZFS mirror. In order to provide a network level redundancy 2 ZVol (virtual block devices as in Figure 1) are created in ZFS which are then managed by DRBD. These 2 block devices are then setup to be block replicated between the 2 hosts over the dedicated network link. Finally the 2 DRBD block devices then host an additional ZFS filesystem which has block-level checksums, compression and nightly snapshots activated.

One of the main challenges of deploying the failover storage for our HV this way was to configure both ZFS and DRBD such that they provide low latency access to the storage. The main changes specific to our setup were to reduce the size of the ZFS write cache as well as increasing the number of commands processed in parallel. The changed that were made to individual parameters were shown in Table 2. The only changes that were made to the default DRBD configuration were to increase buffer sizes for network communications as well as using only MD5 checksums for verifying network communications.

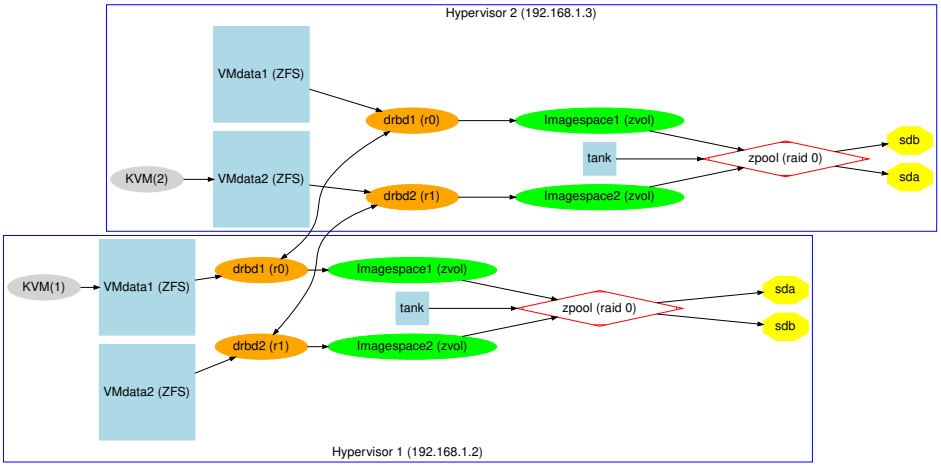

Figure 2. The design of the redundant storage used in the HV2 KVM instances. The physical storage is comprised of 2 disks in each HV which have their content mirrored using ZFS. DRBD then provides network duplication between the 2 HVs and hosts a ZFS filesystem containing the VM image data.

Parameter	default value	optimized value	Description
`zfs_dirty_data_max`	variable	1073741824	This is the size of the write cache in bytes of ZFS. The default of which dynamically changes based on RAM available.
`zfs_delete_blocks`	20480	0	Files larger than `zfs_delete_blocks` × block size are removed synchronously.

Table 2. Table of the list of parameters which were tuned for combing ZFS with DRBD to build a network redundant storage solution.

It was found that after building the HV with both ZFS and DRBD the maximum throughput observed was ~100MB/s. With this in mind cgroups were used on the HV to manage that no single activity exceeded 80MB/s. Adding this additional restriction to the rate at which data could be accessed by a single process reduced the impact that one service could have on the site due to periods of high load.

5 Conclusions

ZFS has proven itself to be a highly flexible and configurable storage technology which can be used for a variety of different workloads. This technology has proven itself to be reliable for maintaining storage at the Edinburgh Tier-2 and is capable of avoiding data loss due to hardware failures.

Additionally ZFS has proven itself as a reliable technology which can be combined with DRBD to provide a highly redundant storage solution for HV systems. The result of all of this work has been that the Edinburgh Tier-2 now hosts all of the required services on 2 HVs which have a high level of redundancy and can withstand the loss of 50% of the servers or 75% of the underlying storage hosting the VMs.

References

[1] ZFS, *A combined file system and logical volume manager*, URL https://zfsonlinux.org/ [accessed on 03/05/2019]

[2] M Ebert and A Washbrook, *Evaluation of ZFS as an efficient WLCG storage backend*, J.Phys.Conf.Ser. **898**, 062054, (2017)

[3] DRBD, *A software-based, shared-nothing, replicated storage solution*, URL https://docs.linbit.com/ [accessed on 03/05/2019]