Zettabyte Filesystem

Vinod .P, Dr. Vijaylakshmi M.N
Student, Associate Professor
Master Of Computer Application
RV college of engineering®
Banglore, India.

Abstract : Data Storage is very critical in storage devices of the computer system as the data stored may be lost in the case of the resource failure, factory defects. The size of the data keeps on increasing day by day, dealing with the larger data the file system must support high access speed and data redundancy. This paper we describes about the file system that provides the high storage space, simple administration, reliability, efficiency, high data integrity and tuning with the regard to physical storage medium using the important design considerations making this possible with few changes in the traditional file system such as the system volume management, snapshots, copy-on-write clones, continuous integrity checking, automatic repair, RAID-Z ,disk scrubbing, resilvering of mirrors, end to end checksums etc. It is a scalable and includes extensive protection against data corruption.

Keywords - System Volume Management, Snapshots, copy-on-write clones, resilvering, Dynamic striping, RAIDZ.

I. INTRODUCTION

Zettabyte file system was developed by Sun Microsystems as a part of a Solaris operating system. It was licensed under the common development and distribution license as an open source file system. ZFS is a 128-bit file system, It can address 18 billion times more data than the 64 bit system. There exists file system like NTFS(NT File System) with several drawbacks like less scalable, low performance, less storage capacity and administration difficulty. But ZFS include the support of volume management, high storage capacities, copy-on-write clones, snapshots, continuous integrity checking .
Zettabyte file system uses the features of copy-on-write technology using which it protects the data by volume management on file system level. When a block of data is altered, it will change the current location on the disk before the new write is finished. If the system crashes or losses power in the process, that data would be lost or damaged. ZFS does not change the location of the data until the write is completed and verified. As it keeps the data safe in case of the system crash.
Zettabyte file system offers additional RAID protection, an regular RAID(Redundant Array of Independent Disks) allows only for two disk failure per volume, whereas in ZFS provides RAID-Z3 which allows a maximum of three disk failures. Zettabyte file system has the ability for setting up multi-disk mirror(nRAID), generally the RAID mirrors are made with a single disk and its copies are stored, but in order to increase the data integrity, the multi-disk mirror provides the multiple copies to increase the read speed. The storage capacity of the regular file system may sooner get problem whereas the ZFS are years ahead in the storage capacity. The possible size of the ZFS Storage pool is 6 EiB = 16 * 2^60 Byte which is as much as 300000 6TB. Creation of new ZFS-pool is very simple, it is mounted automatically and can be accessible immediately. As the disks are added to the storage pool ZFS immediately begins to allocate blocks from those devices and it increases effective bandwidth as each device are added, hence system admin need not to monitor the storage devices to check is the input and output are balanced.

II. SYSTEM DESIGN

The system design contains a client, server and the switch in order to connect the client and the server. The server side contains the gigabyte network Ethernet card connected to the switch. Proxmox 4.1 is installed on the server inside the hard disk of 120GB capacity. Storage pool can be defined as the aggregate collection of the storage devices in order to share the storage spaces. The storage pool defined as the virtual device formed with single disk or the multiple hard disk.
Mirror mode is configured when the system has the more than one or many hard drives on one virtual devices like RAIDZ, RAID-Z2 OR RAID-Z3 to provide the data redundancy. The Zettabyte file system can be composed of both one or several virtual devices. Here we consider the file system in the storage pool is composed of only 1 virtual device and the laptop is used as a client computer with the windows operating system installed on 128GB SSD card .

III. TECHNIQUES BEHIND ZETTA BYTE FILE SYSTEM

A. Storage pools

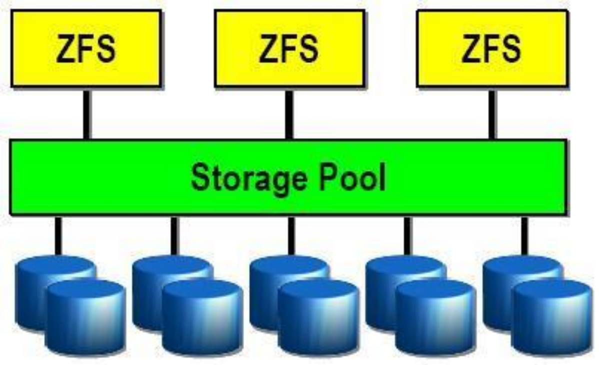

figure 1: storage pool

In traditional file system single storage disk and the volume manager was used. But the zettabyte file system makes use of the virtual storage pools known as the Zpools. Where Zpools is a virtual devices constructed of block devices like files, hard drive partitions etc. Based on the need and the space availability like RAIZ-0 which is non-redundant, as a mirror of two or more devices like RAID-1, as a RAID-Z when there are more devices exists. Apart from the standard storage, the storage devices can be designated as voltaic read caches, non-voltaic write cache or as a spare disk used in the case of failure. So as the time of the mirroring these block devices can be grouped accordingly to physical chassis, hence file system can continue to face the failure of an entire chassis.
Storage pool arrangements are not limited for the similar kind of devices and also contains the heterogeneous collection of device, Zettabyte file system continuously pools them together to provide space for the file system when they are needed and strong devices can be added to the existing file system based on the need of size at any time. When devices like SSD drive which are high speed drives the ZFS will use the SSD as a cache with in the pool. It directs the more frequently used to the SSD and the less frequently used to the other storage device. In Zettabyte file system enables to set the limit of space in file system that instance can occupy and other space can be reserved to be available for file system instance.
These arrangements in ZFS will overcome the drawbacks and increase the read and write speed, increase the storage space and efficiently use the space and also balances the inputs and outputs.

B. Copy-On-Write Transactional Model

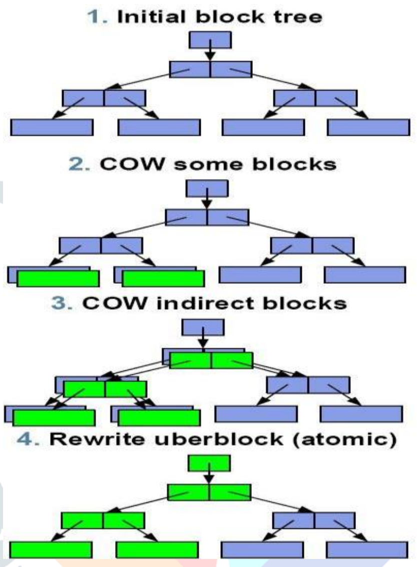

figure 2: copy-on-write transactional model

.

In this transactional model when write operation is to be performed, then the blocks that contain the original data are not immediately over-written, but instead the new blocks are allocated, the data that is needed to be over-written are stored in the newly allocated blocks and if any metadata blocks referencing it will be similarly read, reallocated and written. When the multiple updates are made then they are grouped in to several transactional groups and the intent logs are used when synchronous write semantics are required. The figure 2 shows the flow of the copy-on-write transactional model

C. Snapshots and Clones

A Clone is a writeable copy or file where the contents of the clone is same as the original dataset that created them. A Snapshot can be defined as a read-only and point-in-time copy of the file system state. Whenever a block of data is changed, the changed data is written to a location on disk different from the original copy, creating a new clone will not consume additional disk space. Clones can be created only from the snapshots, when the snapshots are cloned by default dependency is created among clone and snapshot. The clone is created somewhere else in the data hierarchy, the snapshot will not be destroyed till clone exists, clone never inherit the property of the dataset from which it was created using the zfs set and zfs get command which can change the properties of the cloned data.

When a new data is written, then the original existing data will be maintained by the snapshot of file system. The snapshot can be created very quickly, then followed by the clones are created, hence the two file systems are shared in different data blocks and they are independent of each other, the new clone will be reflecting the changes, Whereas the snapshots will still be continued to be shared.

ZFS Clone can be created using the zfs clone command and followed by the path from which the clone has to be created and the name of the new clone. Similarly, the zfs clone can be destroyed using the command zfs destroy.
Example:
```
# zfs snapshot tank/ws/gate@yesterday
# zfs clone tank/ws/gate@yesterday tank/home/bug123
# zfs destroy tank/home/bug123
```

D. Encryption

In Zettabyte file system the data is encrypted using AES (Advanced Encryption Standard) with the key length of 128, 192 and 256 in the CCM and GCM operation modes. It uses the oracle Solaris cryptographic frameworks which gives its access to any available hardware acceleration or optimized software implementations of the encryption algorithms automatically. In addition, a wrapping key is used to encrypt the actual data encryption keys. During the write operation the data blocks are compressed, encrypted, check summed and then duplicated in that order. The encryption policy is defined at dataset level during the creation. The user makes use of the wrapping keys for the change at any time.

E. Dynamic Striping

Dynamic striping means adding the additional devices in the Zpool in order to maximize throughput, hence the write load are balanced among all the disks in a pool, the strip width are not created at allocation time, whenever the pool gets added with a new device, gradually the zettabyte file system will increase the performance by allocating the data to the new devices and within the same pool supports combining different types of virtual devices.

In Zettabyte file system variable sized blocks are used that allows the admin to tune maximum block size as some workloads will not perform good with larger block.

F. Resilvering and Scrub

Resilvering is a process where the data from one device is moved to another device. Because when the entire drive is being replaced by another drive, it will increase the time period and it will depend on the amount of data and size of the drive.

In file systems earlier used the resilvering at the block level. In zettabyte file system the resilvering process is done in controller manner and more powerful than earlier file systems like instead of resilvering of entire data in the disk, zettabyte file system will resilver only minimum amount of the necessary data and entire disk is resilvered only when necessary. The time taken for resilvering of the data is directly proportional to the data size. In zettabyte file system 500Gbytes can be replaced with in few seconds. Resilvering in zettabyte file system is interruptible as whenever there is a power loss, the resilvering process will resume automatically and will be continued, here no manual involvement is required hence resilvering in zettabyte file system is safe.

Scrub is the repair tool used in the zettabyte file system, similar to FSCK (File System Check) which is commonly used in Unix file system, FSCK can be run only when the file system is in offline mode that is mounted, but whereas in zettabyte file system scrub needed not to be in the offline mode and can be used in mounted, the FSCK will always check the metadata, therefore there can be some corrupt in the actual data, whereas scrub check both the metadata and also the actual data.

G. Cache Management

A layer is made in between the main memory and disk with the cache devices that gives the high performance for the read workloads, During the creation or after creation the single or multiple cache devices can be added in the pool. Once the cache devices are added, those cache memories will be filled by the main memory.

Zettabyte file system makes use of the ARC (Adaptive Replacement Cache) and it is a read cache method where as in earlier file system virtual memory page cache were used and for the write purpose it makes use of ZFS Intent log (ZIL), In zettabyte file system both the methods are incorporate as independent virtual devices, “cache” virtual device is for read and “log” virtual device is for write operation.

IV. RAIDZ (REDUNDANT ARRAY OF INDEPENDENT DISKS)

Data integrity is one of the important feature of the zettabyte file system, in order the ensure data integrity the data copies are stored in multiple disks and this technique is known as RAID controller.

RAID-Z is a data distribution schema were each and every block forms its own RAID stripe and it is independent of the block size, here the write hole error can be eliminated when RAID-Z is combined with the copy-on-write semantics and RAID-Z is comparatively faster than the RAID-5 as it is now not needed to perform the usual read-modify-write sequence. It is impossible to determine the RAID-Z geometry as the RAID-Z reconstruction has to traverse the entire metadata of the file system and this can be made possible by integrating the physical and logical structure of the data. When reading a RAID-Z block then RAID compare it with the checksum and if it is found that it returns the wrong values, it finds the block that which returns the bad data and request it to resend the correct data, hence it also detects and corrects silent data corruption.

In Zettabyte file system make use of SHA-256 for entire file system tree where each and every block is check summed and that value is not stored directly in to the block but instead pointer pointing the block is created and stored in that pointer, Again the same pointer is check summed with the value stored within that pointer and this keeps continuing till the node and forms a merkle tree and zettabyte file system stores check summed value in its parent block pointer. If a block is to be accessed, then the checksum is performed and if the value of the checksum gets matches with the stored checksum, then the data in the blocks will be sent to programming stack for the processing, If the checksum value doesn’t match, then zfs provides data redundancy where multiple copies of data are stored and it will fetch those undamaged copy and perform checksum and provide the required data from the block.

CONCLUSION

The file system that is been used at present has various drawbacks such as difficulty in administration, insufficient storage, data integrity issue and to detect the data corruption. But zettabyte file system is a 128bit file system with the pooled storage concepts and includes the various technique like Copy-On-Write transactional model, Snapshots and clones, Encryption, Dynamic Striping, Resilvering and scrub, RAIDZ etc. With the implementation of dynamic striping the file system is fast, data security is ensured using the advanced encryption standard, the pooled storage has increased the storage capacity, implementation of the zettabyte file system is comparatively simple and new standard are being introduced for data integrity and administrating the file system.

REFERENCES:

[1] Himanshu Mishra , Shambhu Bhardwaj, “**Zettabyte File System**”, International Conference on Advanced Computing (ICAC-2016) College of Computing Sciences and Information Technology (CCSIT) ,Teerthanker Mahaveer University , Moradabad ISBN-978-93-5288-834-3, 2016

[2] Eko D. Widianto, Agung B. Prastijo and Ahmed Grifroni, "**On The Implementation of ZFS (Zettabyte File System) Storage System**", 3rd International Conference on Information Technology, Computer and Electrical Engineering, 2016.

[3] Ohad Rodeh, Avi Teperman,"**ZFS - A Scalable Distributed File System Using Object Disks**", IBM Labs, Haifa University, Mount Carmel, Haifa 31905, Israel.

[4] David A Patterson, Garth Gibson, and Randy H Katz, "**A Case for Redundant Arrays of Inexpensive Disks (RAID)**", Department of Electrical Engineering and Computer Science, University of Cabforma Berkeley. CA 94720.

[5] Faheem Hafeez, "**Role of File System in Operating System**", International Journal of Computer Science and Innovation vol.2016 pp-117-127 ISSN: 2458-**6528**

[6] Yupu Zhang, Abhishek Rajimwale, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau," **End-to-end Data Integrity for File Systems: A ZFS Case Study",** 8th USENIX Conference on File and Storage Technologies, San Jose, USA, February 23-26, 2016

[7] Cisco UCS Servers RAID Guide, San Jose: CiscoSystem, Inc., 2015.

[8] Study Yupu Zhang, Abhishek Rajimwale, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau, "**End-to-end Data Integrity for File Systems: A ZFS Case**", Computer Sciences Department, University of Wisconsin-Madison.

[9] Zhi Qiao, Song Fu,"**Charecterizing Declustered Software RAID for Enhancing Storage Reliability and Performance**", The international Conference for High Performance Computing Networks, Storage and Analysis 2018.

[10] Lanyue Lu, Andrea C, Remzi H, "**A Study of Linux File System Evolution, Computer Science Department**", University of Wisconsin, Madison.

[11] Devyani Gurjar, Satish S Kumdhar," **File I/O Performance Analysis of ZFS and BTRFS over ISISCI on a storage pool of Flash Drives**", International Conference on Communication and Electronics System 2019.

[12] Rick Mohr, Adam P Howard, "**Provisioning ZFS Pools on Lustre**", Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines 2019.

[13] Micheal J Brim, Josh K Lothian, "**Evaluating Dynamic File Striping for Lustre**", International Workshop on the Lustre Ecosystem, Challenges and Opportunities, At Annapolis, Maryland.

[14] Gang Hu, **"Study of file encryption and decryption system using security key**", 2nd International conference on computer engineering and technology.

[15] Sarita Kumari, "**A research paper on Cryptography Encryption and Compression Techniques**", International Journal Of Engineering and Computer Science, volume 6, 2017.