

# NLSS: A Near-Line Storage System Design Based on the Combination of HDFS and ZFS

Wei Hu<sup>a</sup>, Guangming Liu<sup>ab</sup>, Yanqing Liu<sup>a</sup>, Junlong Liu<sup>a</sup>, Xiaofeng Wang<sup>a</sup>

<sup>a</sup> College of Computer, National University of Defense Technology, Changsha, China

<sup>b</sup> National Supercomputer Center in Tianjin, Tianjin, China

{huwei, liugm, liuyq, liujl}@nscj.gov.cn, xf\_wang@nudt.edu.cn

**Abstract**— Through analyzing the storage system requirements of supercomputer this paper designs a near-line storage system called NLSS based on the combination of HDFS (Hadoop distributed file system) and ZFS (Zettabyte file system). NLSS uses fat storage nodes (large storage servers) to build near-line storage clusters based on HDFS, and uses the ZFS file system to further enhance HDFS. NLSS effectively reduces the burden of supercomputer online storage system. Experiment results show that NLSS can acquire better storage utilization, reliability and scalability while ensuring appropriate performance.

**Keywords**— dynamic management, HDFS, near-line storage, reliability, ZFS

## I. INTRODUCTION

With the rapid development of supercomputers, its peak computing capacity has reached tens of Pflops/s. Supercomputer provides an important platform which supports the scalability of large-scale scientific parallel applications. In turn, larger and more accurate parallel applications promote the development of supercomputers. Especially the data-intensive applications are becoming an important part of all scientific applications, including oil exploration data processing, genetic and biomedical research, aerodynamics, weather forecasting and climate prediction, numerical simulation of marine environment, new materials development and design etc. This paper presents NLSS which is a near-line storage system based on the combination of HDFS and ZFS. The goal is to achieve a near-line storage system with high space utilization, scalability and reliability which can be an effective complement of supercomputer primary storage system. This work focuses on near-line storage system related techniques, and the data migration between primary storage system and near-line storage system is introduced by another work.

The main contributions are as follows:

- 1) This work presents NLSS near-line storage system which extends the overall capacity of storage systems at lower costs through establishing a hierarchical storage system.
- 2) NLSS is designed to combine HDFS and ZFS with better scalability by making full use of HDFS's horizontal scalability and ZFS's vertical scalability.
- 3) Through using both multi-copy and soft RAID mechanisms, NLSS gets good storage space utilization while ensuring data reliability.

4) Through the experiments on NLSS prototype, we analysed the system performance characteristics under different circumstances and presented the performance optimization suggestions.

The remaining part of this paper is organized as follows. Section II analyses the related technologies and work. Section III introduces the framework of NLSS. Section IV gives the tests of NLSS prototype. Finally, in section V we make some summaries and discuss some future works.

## II. RELATED WORK

Near-line storage [1] is an intermediate type of data storage between online storage and offline storage. The data which will not be used in the near future or have the lower access performance requirements, will be stored on near-line storage. So, near-line storage usually has large capacity, low cost and acceptable I/O performance to meet the needs of the applications or data migration.

HDFS [2] is an open source implementation of GFS (Google's distributed file system), which is a distributed file system based on commodity hardware. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput to application data and good scalability through scaling dynamically.

As shown in Figure 1, ZFS [3] file systems are built on top of virtual storage pools called zpools which are different from traditional file systems. A zpool consists of one or more vdevs. Each vdev can be viewed as a group of hard disks (or partitions, or files, etc.). ZFS ensures data reliability by using its RAID-Z schemes. ZFS is a 128-bit file system [4], so it can address  $1.84 \times 10^{19}$  times more data than 64-bit systems. In

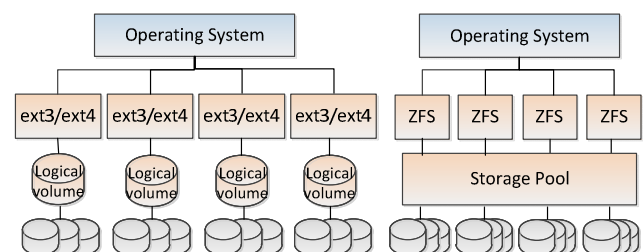


Figure 1. ZFS vs. Ext3/Ext4

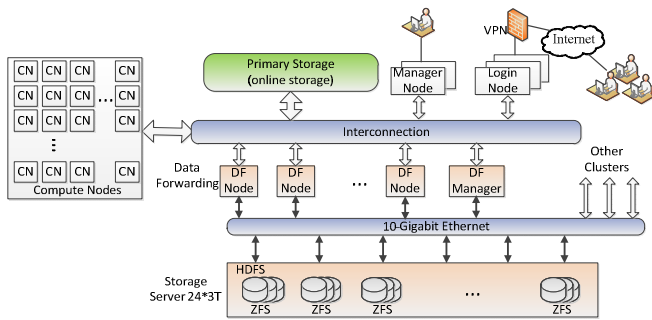


Figure 2. NLSS Architecture

addition, it has many features such as protection against data corruption, efficient data compression, snapshots and copy-on-write clones, continuous integrity checking and automatic repair and so on, which all can be used for large scale near-line storage system.

HPSS (High Performance Storage System) [5] was developed by IBM and DOE National Laboratories, whose goal was to produce a highly scalable high performance storage system. HPSS can manage petabytes of data and provide scalable hierarchical storage management that keeps recently used data on disk and less recently used data on tape.

Tiered Adaptive Storage (TAS) [6] was developed by Cray to provide an open and capacity-optimized data management system. It is designed to reduce the cost of managing storage over the long term and provide tiered storage solutions.

### III. NLSS ARCHITECTURE

The architecture of NLSS is shown in Figure 2. NLSS combines HDFS and ZFS to build the storage cluster using fat storage server. NLSS uses HDFS as a middle tier to organize the whole storage system. It provides data access interface for primary storage system, and manages several ZFS pools on the lower level. Based on HDFS management schemes, NLSS provides good horizontal scalability.

The lower tier uses ZFS file system to manage storage devices which replaces the traditional ext4 file system. It builds shared storage pools by creating RAID-Z redundant space. As the figure 3 shows, using ZFS to manage storage devices for HDFS can not only provide high resource utilization, but also improve the system flexibility.

The reliability of NLSS is also determined by two tiers. The HDFS tier provides reliability by multi-copy mechanism; while the lower tier ZFS uses RAID-Z mechanism. Multi-copy mechanism should follow two principles in the location choice of copy data. One is to keep different copies storing on different racks, and select the storage node from the nearest rack which can improve the storage performance. The other is to keep each storage node's load balance, which can improve storage efficiency. The main issue of multi-copy is the low storage space utilization.

ZFS use RAID-Z to protect data in the lower tier. RAID-Z is a soft RAID scheme using erasure codes. It uses dynamic stripe width: every block is its own RAID stripe resulting in every RAID-Z write being a full-stripe write. This can eliminate the write hole error combined with the copy-on-

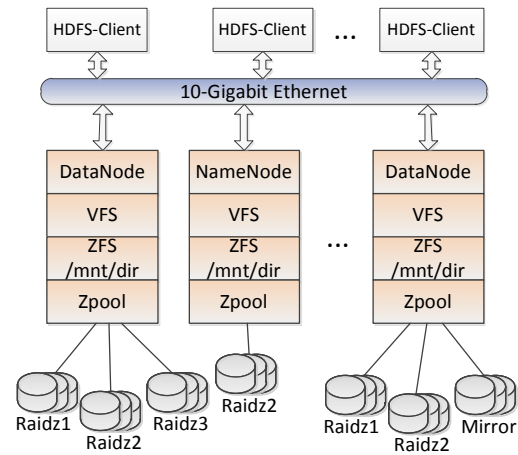


Figure 3. The tiers of NLSS

write transactional semantics of ZFS [7], [8]. RAID-Z is also faster than traditional RAID 5 because it does not need to perform the usual read-modify-write sequence. RAID-Z can not only handle whole-disk failures, but also detect and correct silent data corruption by offering self-healing data which are checksum data.

There are three different RAID-Z modes: RAID-Z1 (similar to RAID 5, allows one disk to fail), RAID-Z2 (similar to RAID 6, allows two disks to fail), and RAID-Z3 (allows three disks to fail). And mirroring is also another ZFS RAID option, is essentially the same as RAID 1, allowing any number of disks to be mirrored.

Therefore, combination of multi-copy and RAID-Z offers a variety of reliable choice to meet the actual needs of different applications, improves system efficiency and flexibility.

### IV. EXPERIMENT

Through building a prototype of NLSS storage system, we obtained the preliminary experiment results about the system performance, reliability and cost. We used 3 redhat Linux servers to establish a NLSS prototype system and another traditional HDFS system based on ext4(referred to traditional HDFS); the details are showed in table 1.

#### A. Storage space utilization

According to the configuration of the server, the 24 hard disks were divided into 3 RAID-Z groups; each group had 8 hard disks. We test the NLSS space utilization combining multi-copy and RAID-Z technologies, as shown in table 2.

Compared with the full redundancy strategy (3 copies) of

TABLE 1. THE DETAILS OF NLSS PROTOTYPE SYSTEM

| Item      | Configuration                           |
|-----------|-----------------------------------------|
| CPU       | 2 x Intel Xeon E5-2640 (6cores, 2.5GHz) |
| Memory    | 24GB                                    |
| Hard Disk | 24*3T                                   |
| Network   | Gigabit Ethernet                        |
| OS        | RHEL6.4                                 |
| HDFS      | Hadoop-1.2.1                            |
| ZFS       | zfs-0.6.2, spl-0.6.2                    |

**TABLE 2.** THE UTILIZATION OF NLSS UNDER DIFFERENT RELIABILITY STRATEGY

| Mechanism (HDFS + RAIDz) | fault tolerance (failure number) | Storage space utilization |
|--------------------------|----------------------------------|---------------------------|
| 2 copies + RAIDZ1        | 2                                | 43.30%                    |
| 1 copies + RAIDZ2        | 2                                | 74.53%                    |
| 1 copies + RAIDZ3        | 3                                | 61.89%                    |

HDFS, the 2 copies with RAID-Z1 can also allow two disks to fail, while the storage space utilization can reach 43.30%. It is 10% more than 3 copies. If the storage reliability strategy use RAIDz2 combined with single copy, the space utilization even nearly double.

### B. Flexibility analysis

The near-line storage system of PB or even EB level in the future for supercomputer requires better scalability. Using HDFS to build massive near-line storage system has good horizontal scalability. When the storage space is insufficient, we can expand the system's available space by adding storage nodes. But HDFS is not good at scaling vertically due to some limitations on expanding the nodes themselves online.

ZFS provides the vertical scalability through storage pool scheme to enhance the scalability of the whole system. We can add and replace the hard disk online at any time, and this can be easily used to disk failure fault-tolerance and space expansion.

In the experiment, we used 5 hard disks to create a RAID-Z group. Then we tested the disk replacement operation online. One disk was pulled out randomly and replaced with a new one using the command "replace zpool". As shown in Figure 4, after the replacement of a new disk, RAID-Z started the data recovery process, and the rate is 177MB/s. ZFS solves the problems of system vertical scalability, and improves the system flexibility.

### C. Read and write performance analysis

NLSS storage system is designed to have a rational I/O performance to meet the requirements of the data transmission or application. NLSS can connect the storage servers through fibre channel for the high bandwidth requirement.

In this test, we used two storage servers connected by Gigabit Ethernet and two pairs of disk groups in each storage server to do the comparison experiment. One group is a pair of

```
[root@Master ~]# zpool status
pool: hadoop
state: DEGRADED
status: One or more devices is currently being resilvered. The pool will
        continue to function, possibly in a degraded state.
action: Wait for the resilver to complete.
scan: resilver in progress since Fri Aug 8 16:02:51 2014
      4.49G scanned out of 5.96G at 177M/s, 0h0m to go
      920M resilvered, 75.35% done
config:

NAME        STATE      READ WRITE CKSUM
hadoop      DEGRADED   0     0     0
raidz1-0    DEGRADED   0     0     0
sdb         ONLINE    0     0     0
sdc         ONLINE    0     0     0
replacing-2 UNAVAIL    0     0     0
  old       UNAVAIL    4     1     0   corrupted data
  sdd       ONLINE    0     0     0   (resilvering)
sde         ONLINE    0     0     0
sdf         ONLINE    0     0     0

errors: No known data errors
```

**Figure 4.** Disk replacement and data recovery

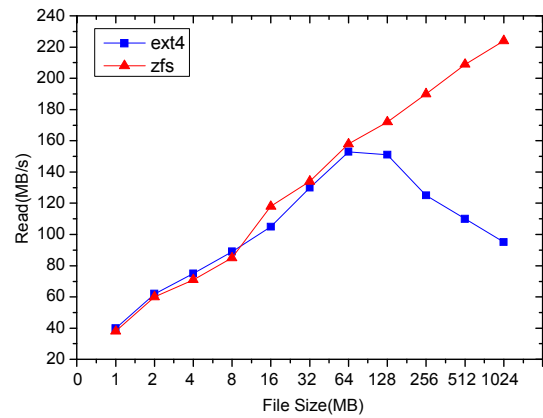


**Figure 5.** Reading test for 10 files concurrently

5 hard disks each with ext4 in two storage servers. Another group is a pair of RAID-Z1 each consisted of 5 hard disks in two storage servers.

The TestDFS10 which was the Hadoop built-in tool was used to test the concurrency throughput of reading and writing. We configured the test task with 8 Mapslots and 8 Reduceslots to complete the throughput test by running 8 Map and 8 Reduce concurrently. So we had the throughput results of reading and writing for concurrent tests as follows.

In the reading test HDFS was configured with two copies, default block size is 64MB, and 10 different files were used for testing. The results of read performance on different file granularity are shown in Figure 5. With the increasement of file size, the read performance of NLSS increases rapidly and linearly, while the traditional HDFS reaches maximum on the file granularity 64MB. As the results shown, if the CPU resources are adequacy, NLSS can support better read performance especially for large files.

In order to analyse the read performance of the system when CPU resources were scared, we increased the test files to 50 which is more than the maximum number of concurrent tasks (12). From the test results shown in Figure 6, it can be found that NLSS has better performance when concurrently reading massive data. If CPU resources are limited, the throughputs of two systems both suffer a decline. Compared to traditional HDFS, when the file size is greater than 8 MB, NLSS has better read performance and better adaptability for much load.

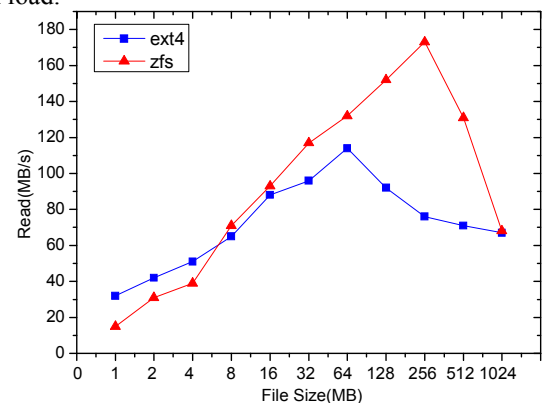


**Figure 6.** Reading test for 50 files concurrently

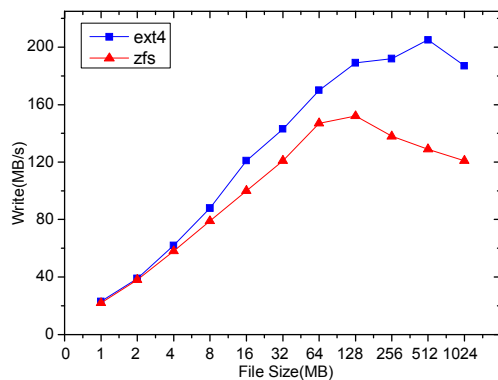


Figure 7. Writing test for 10 files concurrently

In the experiment above, the system read performance was constrained by Gigabit Ethernet. In order to test the write performance of the systems, the HDFS tier was configured with one copy to avoid the influence of network bandwidth shortage.

Figure 7 shows the write bandwidth of the two systems for 10 different files concurrently. The concurrent write performance of NLSS is not as good as traditional HDFS when CPU resources are adequate. We have the analysis that NLSS which using ZFS stripes the data before writing them to disks. This is a non-negligible overhead which will be even bigger with the file size increasing.

In order to study the effect of CPU resource on the write performance, we test the write bandwidth when system writes 50 GB data in different file granularities, as shown in Figure 8. In the case of insufficient CPU resources, when the file size is less than 256MB, the performance of NLSS is better than traditional HDFS. With the file size increasing, the number of concurrent tasks reduces, and the write bandwidth of NLSS decreases, while the bandwidth of traditional HDFS increases.

From a series of reading and writing tests, the performance of NLSS is affected by the file size, system resources, network bandwidth, and so on. Some analyses are as follows:

1) For NLSS or traditional HDFS, the performance is affected by the block size of HDFS. The system performance can benefit from setting different block sizes based on the user data's characteristics.

2) For NLSS, RAIDZ striping of ZFS consumes some system resources, and the multi-directory management also consumes partial system resources. In the case of sufficient CPU resources, compared to ZFS striping, the overhead of multi- directory management has smaller impact on the system performance, as the Figure 7 shows. In the case of insufficient CPU resources, the overhead of multi-directory management has a bigger impact on the system, as the Figure 8 shows.

3) The NLSS system based on the ZFS soft RAID can improve concurrent read performance of the whole system.

## V. CONCLUSION

This paper designs a massive near-line storage system NLSS for supercomputer. Based on the requirements analysis, we establish the near-line storage system based on HDFS. Due

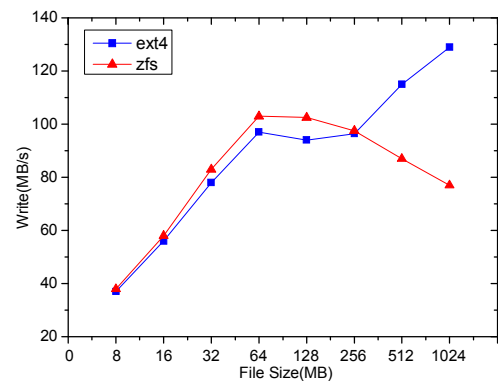


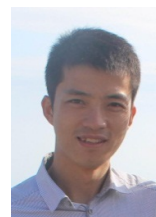
Figure 8. Writing test using file of 50 GB size

to the low space utilization and limitation of vertical scalability of HDFS, this work proposes NLSS storage system based on HDFS and using ZFS to replace the traditional ext3 / ext4 file system. Through the experiments and analyses of the NLSS prototype system, it has better reliability and scalability, high space utilization, better concurrent read performance and rational write performance for the specific user data. NLSS provides a reliable solution on the near-line storage for the high performance computing and other computing systems.

In the future, our efforts will mainly focus on optimizing NLSS system performance to meet a variety of needs and completing the tests of data migration module.

## REFERENCES

- [1] *Nearline storage*. [https://en.wikipedia.org/wiki/Nearline\\_storage](https://en.wikipedia.org/wiki/Nearline_storage).
- [2] *Apache Hadoop*. <https://hadoop.apache.org/>.
- [3] Solaris Zfs. Administration Guide, April 2009. White paper.
- [4] J. Bonwick and B. Moore. ZFS: The last word in file systems. 2007.
- [5] R. Watson and R. Coyne. "The parallel I/O architecture of the high-performance storage system (HPSS)". in *Proc. Fourteenth IEEE Symposium on Mass Storage Systems*, 1995, pp. 27-44.
- [6] *Cray Tiered Adaptive Storage (TAS)*. <http://www.cray.com/products/storage/tiered-adaptive-storage>.
- [7] A. Kadav, A. Rajimwale. Reliability Analysis of ZFS. <http://pages.cs.wisc.edu/~kadav/zfs/zfsrel.pdf>, 2010.
- [8] Y. Zhang, A. Rajimwale, A. Arpaci-Dusseau, et al. "End-to-end Data Integrity for File Systems: A ZFS Case Study". in *Proc. FAST*, 2010, pp. 29-42.



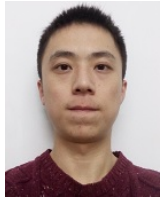
**Wei Hu** received the B.S. degree from PLA University of Science and Technology, China, in 2004, and the M.S. degree from National University of Defense Technology, China, in 2010. He currently pursues the Ph.D. degree in the College of Computer, National University of Defense Technology, Changsha, China. His research interests include high performance computing and machine learning.



**Guangming Liu** received the B.S. and M.S. degrees from National University of Defense Technology, China, in 1980 and 1986 respectively. He is now a professor in the College of Computer, National University of Defense Technology. His research interests include high performance computing, massive storage and cloud computing.



**Yanqing Liu** received the B.S. and M.S. degrees from National University of Defense Technology, China, in 2012 and 2014, respectively. He is now an assistant engineer in the College of Computer, National University of Defense Technology. His research interests include high performance computing and massive storage.



**Junlong Liu** received the B.S. degree from National University of Defense Technology, China, in 2013. He currently pursues M.S. degree in the College of Computer, National University of Defense Technology. His research interests include high performance computing and massive storage.



**Xiaofeng Wang** has been working as an assistant professor in the College of Computer at National University of Defense Technology in China. He received the B.S., M.S. and Ph.D. degrees in computer science from National University of Defense Technology in 2004, 2006 and 2009 respectively. His research interests include trustworthy networks and systems, applied cryptography, network security.