International Journal of Database Theory and Application
Vol.7, No.3 (2014), pp.91-106
http://dx.doi.org/10.14257/ijdta.2014.7.3.10

Research and Implementation of Distributed Disaster Recovery System Based on PRS Algorithm

Jian Wan[1, 2], Huijia Xuan[1, 2] and Jilin Zhang[1, 2, 3]

[1]School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China
[2]Key Laboratory of Complex System Modeling and Simulation, Ministry of Education
[3]Electric Engineering School, Zhejiang University, Hangzhou, China
[1]yongjian.ren@infocore.cn, [2]cloudxhj@gmail.com, [3]jilin.zhang@hdu.edu.cn

Abstract

With the advent of the information age, the data volumes of various industries are in explosive growth, which increasingly highlights the importance of data. When a disaster occurs, how to recover the data completely and rapidly is one of the important issues receiving common concerns throughout the current industrial and academic circles. Based on the traditional disaster recovery system, this paper researches and implements a PRS algorithm-based distributed disaster recovery system[1, 2] called OsnDDR, which ensures the high availability of the system under the circumstance of continuous disasters or multi-node damage. The system adopts PRS algorithm based on RS erasure code [3] to reduce the storage resource consumption caused by the data redundancy and designs the load balancing strategy to guarantee the load balancing of each disaster recovery center in the system.

Keywords: *Disaster recovery system, Distributed, Erasure code, Load balancing*

1. Introduction

Currently, the common structure of the disaster recovery system is one-to-one or multi-to-one, that is, one production center corresponds to one disaster recovery center, or multiple production centers correspond to one disaster recovery center [4]. In the case of single point failure, the above structure can generally ensure the data security, but if this structure encounters multi-node failures, it can't ensure the data security and business continuity. Symantec's Veritas Volume Replicator [5] system allows one-to-multi structure to ensure the data security, but it has the disadvantages of big data redundancy and low storage resource utilization. Meanwhile, as most disaster recovery products on the current market are based on expensive disk arrays and FC (Fiber Channel) network and integrated in hardware devices, they have certain defects on flexibility, procurement cost, soft hardware universality, etc. One example is EMC's SRDF [6] disaster recovery system, which is required to be built on top of the Symmetrix storage system, and thus lacks the support for heterogeneous storage systems; NetApp's SnapMirror [7] system is a disaster recovery system based on COW [8] (copy on write) technology with relatively good performance, but the system relies on NetApp's storage devices and WAFL [9] file system. In view of the above conditions, this paper presents a distributed disaster recovery system based on the structure of 1+1+N with low dependency. To construct a 1+1+N structured disaster recovery system, it needs to solve the following technical issues: production and management of the redundant data, data recovery after disasters, load balancing between disaster recovery centers, and storage resource utilization rate, etc. Considering of the above problems, this paper designs and realizes a load balancing strategy and puts forward the PRS algorithm to ensure data security and improve the utilization rate of storage resources and system performance.

This paper will organize the content as follows: Section 2 discusses the architecture and treatment scheme of OsnDDR, as well as the design of I/O management module. Section 3 presents disaster recovery center's performance model and load model, and the load balancing strategy suitable for OsnDDR. Section 4 introduces the principle and handling process of PRS algorithm. Section 5 tests the reading and writing performance of the application and the encoding time of RS erasure code. Section 6 gives the conclusion.

2. OsnDDR Architecture

This paper designs a distributed disaster recovery system named OsnDDR based on the storage virtualization engine. Storage virtualization engine manages underlying heterogeneous storage resources by the storage virtualization technology [10-12] and maps storage resources in the form of logical volumes to servers via iSCSI [13] (Internet small computer system interface) or FC network. According to user demand, OsnDDR establishes a hot backup center [14] in remote or only on local important data for disaster recovery backup. OsnDDR has no strict requirement on transmission network and does not need specialized communication link and hardware facilities, thereby reducing the cost of system deployment.

2.1. System Architecture and Function of each Module

OsnDDR is mainly composed of four function modules, including I/O management module, file transfer module, load balancing module, encoding and decoding module. These modules are respectively deployed in the nodes with different functions which include production center, disaster recovery center, and control center. The specific deployment is shown in Table1.

Table 1. Function Modules and Deployment Position

Function Module	Position
I/O Management Module	Production Center
File Transfer Module	Production Center and Disaster Recovery Center
Load Balancing Module	Control Center
Encoding and Decoding Module	Disaster Recovery Center

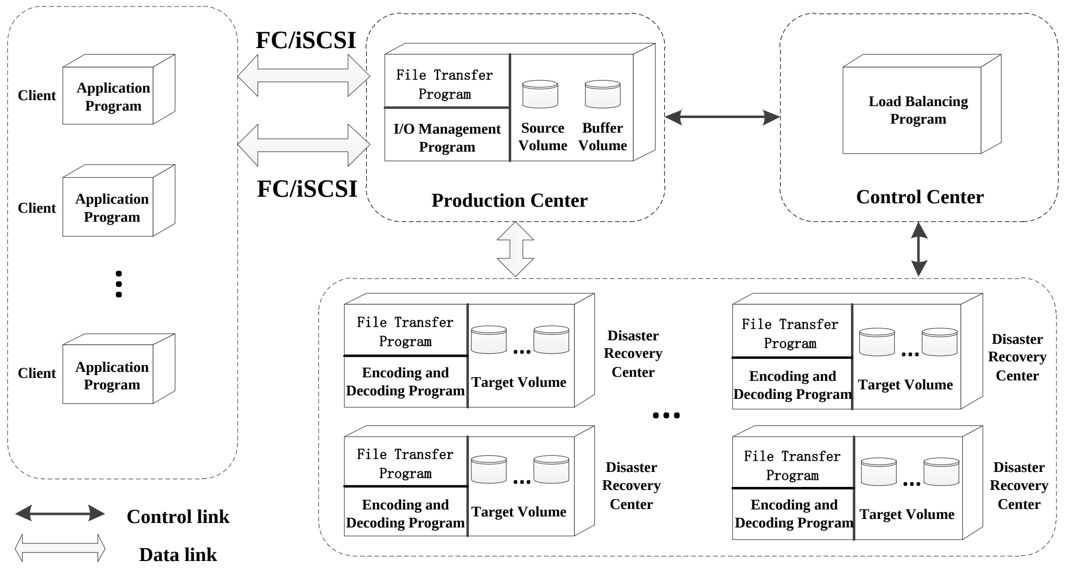

Figure 1. Architecture of OsnDDR

The overall architecture of OsnDDR is shown in Figure 1. The production center maps the logical volume (source volume) to the client for application, stores the client's source data, and configures an asynchronous buffer volume for the source volume. The control center implements the load balancing strategy to improve the load balancing of OsnDDR. The disaster recovery center implements PRS algorithm and stores the redundant data generated by RS erasure code in the local logical volume called target volume.

I/O management module is deployed in the production center, which can access the data of I/O request packet (IRP) written in source volume by the filter driver [15-17] in OsnDDR (DDR) and stores the data of IRP in the buffer volume through the asynchronous transfer mode [18].

File transfer module is based on Web Service [19, 20] technology and deployed in the production center and disaster recovery center. This model is used to transmit data, including source data and redundant data, and it reduces the amount of data transmission through the data compression technology.

Load balancing module is deployed in the control center. This module divides the disaster recovery center into different groups according to the geographical location of disaster recovery center, and builds the performance model and load model of the disaster recovery center. Based on the above three points, the load balancing module selects the optimal disaster recovery center.

Encoding and decoding module is deployed in the disaster recovery center. When the disaster recovery center receives source data, the encoding module will parse these data and encode every IRP by RS erasure code, and then store the redundant data generated by RS erasure code in multiple files for transmission. The decoding module is responsible for the recovery of source data when multiple nodes are destroyed based on the redundant data.

2.2. Process of OsnDRR

The process of OsnDRR mainly includes disaster recovery relationship configuration, relationship initialization, data protection, and data recovery.

(1) disaster recovery relationship configuration

Disaster recovery relationship configuration refers to the process of deploying multiple target volumes into a source volume. The number of disaster recovery centers to be deployed should not exceed the one in OsnDDR. Assuming that there are N disaster recovery centers in OsnDDR, and users deploy relationship with (K, M) ($M<=N$) mode for a source volume. (K,M) mode means cutting the source data with the length L into K parts, generating M redundant data with the length L/K after encoding by RS erasure code and saving them in M target volumes. Load balancing module in the control center will receive this configuration information and then select M disaster recovery centers with the best ratio of performance and load for this relationship.

(2) relationship initialization

After configuring a relationship, we must initialize it to ensure the data consistency between source volume and target volumes. What relationship initialization does is encoding all data in source volume by RS erasure code and saving them into target volumes based on the disaster recovery relationship.

(3) data protection

After the step 2, DDR monitors all IRP which are written into the source volume and then stores the source data in buffer volume. When the amount of source data in buffer volume reaches a certain value, file transfer module will read the source data and send the control information to the control center to get the address of disaster recovery center which has the best ratio of performance and load, and then the source data will be compressed and sent to the corresponding disaster recovery center.

File transfer module in the disaster recovery center receives the source data and sends it to the encoding and decoding module by I/O control (IOCTL). The source data will be encoded by RS erasure code. Massive dense matrix computing in RS erasure code results in bad performance, thus OsnDDR uses parallel mode [21, 22] to improve the efficiency. After the completion of encoding, the redundant data will be sent to target volumes for storing by file transfer module according to the disaster recovery relationship.

(4) data recovery

When no more than M-K disaster recovery centers are damaged in the (K, M) mode, OsnDDR can take the advantage of the surplus redundant data and RS erasure code to construct the source data. And then according to the recovered source data, OsnDDR can recover the data in the damaged disaster recovery center.

2.3. Design of I/O Management Module

DDR is the core component of OsnDDR, which is a filter driver deployed on the logical volume device and implements I/O management module and encoding and decoding module. DDR controls every IRP which is written into the source volume and stores IRP’s data into the buffer volume by the asynchronous transfer mode. File transfer module sends the source data in the buffer volume to the corresponding disaster recovery center for encoding by RS erasure code. When multi-node damage including production center, we should map residual target volumes to the new production center and send IOCTL to DDR, which will read the data in each target volume and carry out decoding operations.

Asynchronous transfer mode of DDR requires a buffer volume which is used to temporarily store all IRP for the source volume. Meanwhile, in order to reduce the impact of

DDR on reading and writing performance of applications, DDR needs to deploy a memory segment for each pair of the disaster recovery relationship. These memory segments store IRP's data temporarily. If the space of segment is insufficient, DDR will write the source data in segment to the buffer volume. Buffer volume is divided into control field and data field. Control field records the space usage of the buffer volume, amount of segments, next segment's offset and so on, while data field stores all segments. The organizational structure of the buffer volume and segment is shown in Figure 2.

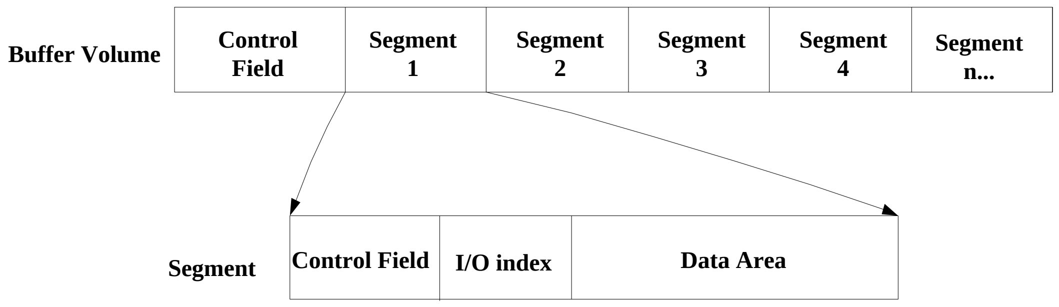

Figure 2. Organizational Structure of Buffer Volume and Segment

Segment is made up of control field, I/O index and data area. Control field stores the amount of IRP, the space usage of segment and so on. I/O index is array of the struct which stores IRP's offset and length in the source volume. Data area stores IRP's data.

3. Load Balancing Strategy in OsnDDR

To ensure the data security in the case of the multi-node damage, OsnDDR changes the traditional one-to-one remote disaster recovery mode into a distributed one, which enables users to configure the amount of the disaster recovery center for a disaster recovery relationship according to their needs. During the process of selecting disaster recovery centers, we should consider the system's performance and keep its load balancing. Furthermore, the disaster recovery center needs to undertake the task of encoding the source data by RS erasure code. Therefore, in order to avoid poor performance resulted from computational load aggregation, it is also important to distribute disaster recovery centers' computational load.

The load balancing of OsnDDR is mainly achieved by two parts. Firstly, select appropriate disaster recovery centers according to the load balancing strategy and group before configuring the relationship. Secondly, select the disaster recovery center which has the best ratio of performance and load for encoding.

The main purpose of the load balancing is to improve the system performance and reduce the execution time by assigning the right amount of load to the server based on its performance [23]. Therefore, in order to achieve the load balancing in OsnDDR, we should firstly build unified models to evaluate the performance and load condition of the disaster recovery center.

3.1. Performance Model of the Disaster Recovery Center

In general, CPU, network bandwidth, disk performance, memory and OS are the main influencing factors on the server's performance [24]. But it is difficult to evaluate the server's performance by the pattern like white-box testing, namely determining the weight of the above hardware and software resources according to their influence to the server's performance. If the weight is improper, the system's load balancing strategy will be broken. However, the disaster recovery centers' performance will have an impact on the logical volume's I/O performance. Therefore, we can use logical volume's reading and writing performance under the same I/O load to express the corresponding disaster recovery centers' performance.

I/O load is related to the number of waiting I/O, the proportion of random I/O and sequential I/O, the ratio of writing I/O and reading I/O and the size of I/O block. The I/O performance can be described by the average response time, throughput rate and the number of writing and reading operations per second [25]. In this paper, we study the relationship between I/O load and I/O performance. From our experimental data, we can see that the number of waiting I/O has a good linear relationship with the average response time, and the disaster recovery centers' different performance result in different slopes between them (shown in Figure 3). This paper uses the slope fitted by the least square method between the number of waiting I/O and the average response time to describe the performance of the disaster recovery center.

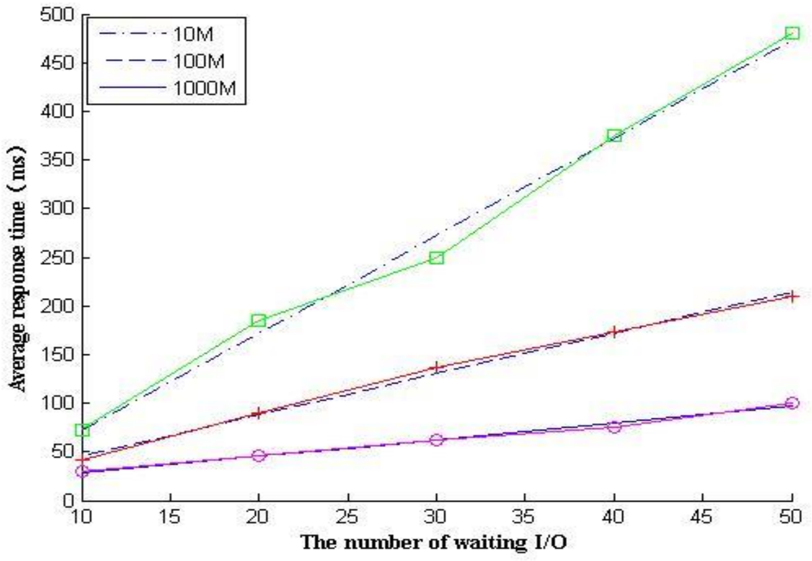

Figure 3. Relationship between the Number of Waiting I/O and Average Response Time

Definition 1 (performance model of the disaster recovery center): There are N disaster recovery centers in OsnDDR, and their corresponding values of the relative performance are $\{P_1,P_2,...,P_n\}$, and the slopes between the average response time and the number of waiting I/O are respectively $\{k_1,k_2,...,k_n\}$.After making $P_1 = 1$,we arrive at

$$P_i = \frac{k_1}{k_i} \times P_1 = \frac{k_1}{k_i}, i = 1,2,..., N \tag{1}$$

3.2. Load Model of Disaster Recovery Center

The main function of the disaster recovery center is to store the redundant data generated by encoding through the target volumes. Meanwhile, the disaster recovery center needs to take tasks of encoding by erasure code. Therefore, for the disaster recovery center, load is mainly composed of writing operation to the target volume and encoding. In OsnDDR, DDR records the load of disaster recovery center, and interacts with the control center of the load information.

Definition 2 (load model of the disaster recovery center): For a disaster recovery center, encoding computational load is C_i, and it has N logical volumes, and writing load of each logical volume is W_k, the total load L_i of the disaster recovery center V_i is

$$L_i = C_i \times \sum_{k=1}^{N} W_k \tag{2}$$

3.3. Load Balancing Strategy in OsnDDR

Definition 3 (performance weighted load model of the disaster recovery center): In OsnDDR, the values of the performance weighted load in N disaster recovery centers are $\{PL_1, PL_2, \ldots, PL_n\}$, we get

$$PL_i = \frac{P_i}{L_i}, i = 1,2,\ldots, n \tag{3}$$

P_i means the performance value of the disaster recovery center i, which can be calculated by the definition 1. L_i is the total load of the disaster recovery center i, which can be calculated by the definition 2.

The control center needs to choose multiple disaster recovery centers when receiving a command to create the disaster recovery relationship. Firstly, the control center quantifies all disaster recovery centers by performance weighted load model. Then, the center selects multiple disaster recovery centers with the optimal performance weighted load value in different groups. Furthermore, the control model just relies on the performance weighted load model to select a disaster recovery center before encoding by RS erasure code.

4. Implementation Principle and Optimization of PRS Algorithm

To ensure the security of source data, OsnDDR adopts the data redundancy backup mechanism and stores the redundant data into several disaster recovery centers. Currently, data redundancy backup mechanism mainly includes two sorts: exact mirror mechanism and erasure code mechanism. Exact mirror mechanism, also known as the copy mechanism, refers to multiply the source data into several copies and store them separately. This mechanism is easily accessible, but has low resource utilization rate, which will lead to a high cost [26]. Erasure code mechanism produces the redundant data by encoding. When a part of the source data is damaged, erasure code mechanism can recover the source data according to the decoding by erasure code. Compared with the exact mirror mechanism, erasure code mechanism has the advantage of high disk utilization and low redundancy [27].

4.1. Optimize basic Algorithm in Galois Field

In OsnDDR, PRS algorithm adopts the RS erasure code. RS erasure code is the erasure code which performs polynomial operations of field element in Galois Field $GF(2^w)$ [28]. In $GF(2^w)$, the definition of the addition and subtraction is equivalent to XOR, and that multiplication and division usually involve computing discrete logarithms and table lookup operation, thus the computation overhead in RS erasure code is high [29].

Since OsnDDR handles the source data in bytes, the operation in $GF(2^W)$ is limited in $GF(2^8)$. Taking GF (2^3) construction as an example to illustrate GF (2^8) construction, the primitive polynomial of GF (2^3) is $P(x) = x^3 + x + 1$, and the root of $P(x) = 0$ is defined as a, then $a^3 + a + 1 = 0$. As addition and subtraction of GF (2^3) are equivalent to XOR, then $a^3 = a + 1$. The calculated GF (2^3) elements are shown in Table 2.

Table 2. Feature Construction in GF(2^3)

Exponential Form	mod(a^3+a+1)	Binary Form
0	0	000
a^0	1	001
a^1	a	010
a^2	a^2	100
a^3	a+1	011
a^4	a^2+a	110
a^5	a^2+a+1	111
a^6	a^2+1	101

As shown in Table 2, the element polynomials in GF (2^3) are one-to-one correspondence to the binary form and exponential form. The same method can be used to construct the 256 elements in GF (2^8).

We take arithmetical operation in GF(2^3) as an example to show the features of arithmetical operation in GF(2^W)

Addition: $a^3 + a^5$=(011) $\oplus$(111)$= a^2$

Subtraction: $a^3 - a^5$=(011)$\oplus$(111)$= a^2$

Multiplication: $a^3 \times a^5 = a^{(3+5)\mathrm{mod}7} = a$

Division: $a^3 \div a^5 = a^{(3-5)\mathrm{mod}7} = a^5$

As can be seen from the above calculating process, the exponential form of GF (2^w) elements is more suitable for multiplication and division, and the binary form is more suitable for the addition and subtraction. To optimize the domain arithmetic, OsnDDR uses a double-table lookup method to alternative basic arithmetic operations. Firstly, we build two tables named Table 1 [i] and Table 2 [i]. Table1 stores the congruent relationship from the exponential form to the binary form, namely i refers to the exponential value and Table 1 [i] refers to the corresponding binary value. Table2 stores the congruent relationship from the binary form to the exponential form, namely i refers to the binary value, Table 2 [i] refers to the corresponding exponential value. Then, addition and subtraction will be directly converted into XOR of the binary form, and that multiplication and division will be converted into the following forms:

X × Y = Table1 [(Table2 [X] + Table2 [Y]) mod (2^8-1)]

X ÷ Y = Table1 [(Table2 [X] − Table2 [Y]) mod (2^8-1)]

Therefore, the double-table lookup method converts multiplication and division in GF(2^W) into three table look-up operations, one addition or subtraction operation and one modulus operation to improve the computing efficiency.

4.2. Parallel Implementation of RS Erasure Code

Figure 4 shows the encoding process of PRS algorithm. To reduce the computing load of the production center, OsnDDR performs PRS algorithm in the disaster recovery center. Once receiving the source data, the disaster recovery center will analyze the data and send the analyzed one to the encoding and decoding module by IOCTL. Then the encoding and decoding module encodes the source data in groups of IRP and writes the redundant data into distributing files. After encoding process is accomplished, file transfer module will transfer the distributing files to the target volumes.

Encoding and decoding module needs to recover the source data when the data is lost. If the data loss occurs in the disaster recovery center and the production center takes no damage, we only need to reestablish the disaster recovery center and recode the source volume according to the configuration file. If the production center is damaged, new production center should be built and the storage virtualization engine should map the remaining intact target volumes to the new production center. The source data will be recovered based on the decoding by RS erasure code and written into the source volume.

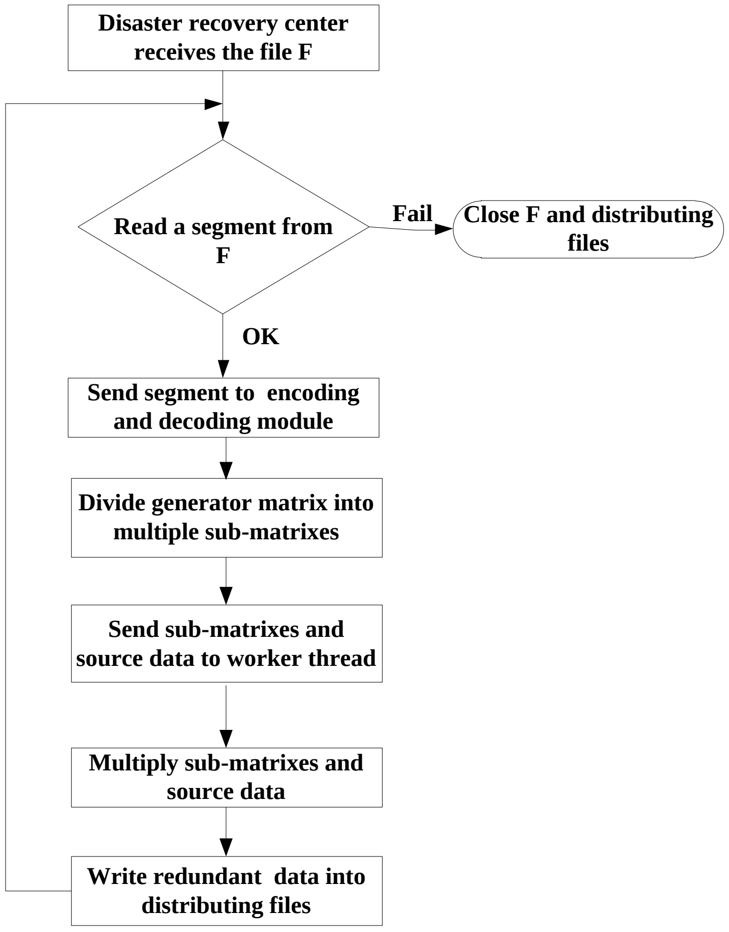
Disaster recovery center receives the file F
Read a segment from F
Fail
Close F and distributing files
OK
Send segment to encoding and decoding module
Divide generator matrix into multiple sub-matrixes
Send sub-matrixes and source data to worker thread
Multiply sub-matrixes and source data
Write redundant data into distributing files

Figure 4. **Encoding Process of PRS Algorithm**

As can be seen in Figure 4, the core of PRS algorithm is the encoding and decoding operation of RS erasure code. The encoding operation makes the generator matrix multiplied by the column vector of the source data to get the column vector of the redundant data, while the decoding operation makes the column vector of the surplus redundant data multiplied by the inverse matrix of its corresponding remaining generator matrix to get the source data [30].

In the encoding and decoding process of RS erasure code, the key step is the matrix multiplication operation. PRS algorithm uses the parallel algorithm to optimize matrix multiplication operation to improve the enforcement efficiency of RS erasure code and system performance. The specific steps are as follows: let $G_{M \times K}$ be the generator matrix, and let $S_{k \times (L/k)}$ denote the source data matrix, we need to solve the matrix $R_{M \times (L/K)} = G_{M \times K} \times S_{k \times (L/k)}$, where K represents that the source data is cut into K parts, L is the length of the source data, M is the copies of redundant data after encoding, and the number of worker thread is M. Firstly, the generator matrix is divided into M sub-matrixes according to the row, and then the sub-matrixes M_i and the source data matrix are sent to worker thread Pi, which makes the sub-matrixes M_i multiplied by the source data matrix. The computed result is stored in different distributing files, which will be sent to the different disaster recovery centers by the file transfer module.

When using the above matrix multiplication algorithm, there is no message communication between threads and each thread processes a separate operation, so the time overhead of the above matrix multiplication algorithm is primarily composed of the time which worker treads get the data and complete the matrix multiplication.

5. Performance Evaluation

5.1. Experimental Environment Description

OsnDDR is deployed under the laboratory environment which simulates the real remote disaster recovery environment by limiting the bandwidth. Experimental environment includes a client, a production center, a control center, and seven disaster recovery centers, where the client, the control center and three disaster recovery centers consist of the virtual machines. Client, production center and control center are in the same gigabit LAN. Production center and control center connect with the disaster recovery centers under 10Mb network. The hardware and software configurations of the physical machine nodes are shown in Table 3, and the configuration of the virtual machine nodes are shown in Table 4.

Table 3. Physical Node Configuration Table

Role	CPU	Memory	Disk	OS
Production Center	Intel Core 2.98GHz CPU*4	4GB	500G	Windows Server 2003
Disaster Recovery Center 1	Intel Xeon 2.13GHz CPU*4	4GB	500G	Windows Server 2003
Disaster Recovery Center 2	Intel Core 2.98GHz CPU*4	4GB	500G	Windows Server 2003
Disaster Recovery Center 3	Intel Core 2.98GHz CPU*4	4GB	500G	Windows Server 2003
Disaster Recovery Center 4	Intel Core 3.10GHz CPU*2	2GB	500G	Windows Server 2003

Table 4. Virtual Machine Configuration Table

Role	CPU	Memory	Disk	OS
Client	2	2GB	100G	Windows Server 2003
Control Center	4	2GB	100G	Windows Server 2003
Disaster Recovery Center 5	2	2GB	200G	Windows Server 2003
Disaster Recovery Center 6	4	4GB	200G	Windows Server 2003
Disaster Recovery Center 7	4	2GB	200G	Windows Server 2003

5.2. OsnDDR's Impact on Application

In OsnDDR, DDR needs to get all IRP which are written to the source volume and perform a series of data alignment operations that have certain effects on the reading and writing performance of applications. In this paper, four groups of the experiments are made to test this effect, and the experiments shows that OsnDDR can guarantee the normal operation of the user's applications.

Firstly, in Gigabit network environment, the source volume in the production center is mapped to the client. The software, called iometer [31], run in the client, reads and writes the source volume to obtain the reading and writing performance without the disaster recovery relationship. Secondly, the source volume is configured on (2, 4) mode disaster recovery relationship, the iometer records the reading and writing performance of the source volume after configuring. The results are shown in Figure 5.

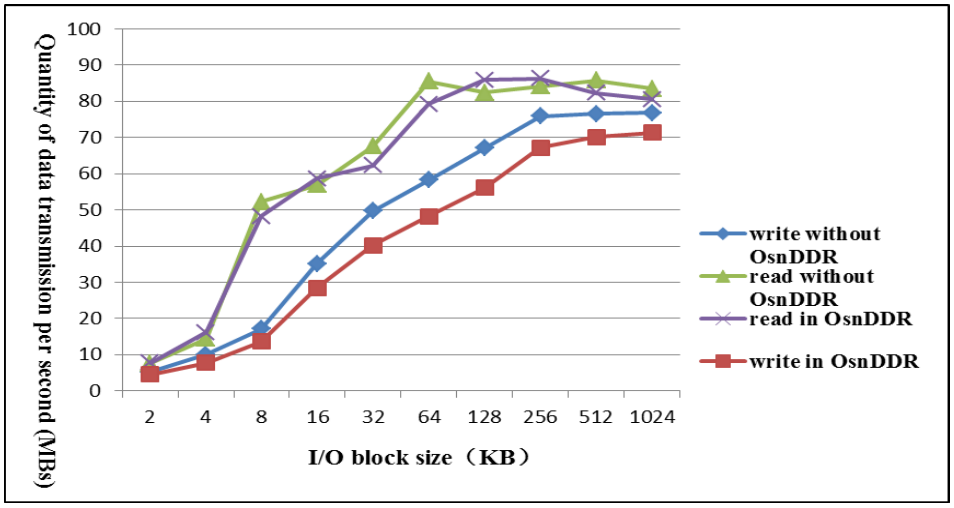

Figure 5. Reading and Writing Performance

According to the Figure 5, the reading performance of the volume is better than writing performance. The configuration of OsnDDR exerts little influence on the reading performance, while the writing performance would be degraded to some extent. There is 15% decline in the average writing performance after deploying OsnDDR, and the ratio is relatively low. Therefore, OsnDDR can guarantee the normal operation of the applications. Besides, this test also proves the feasibility of OsnDDR.

5.3. Implement Performance Comparison of RS Erasure Code

This group of experiments presents the performance testing and comparison of serial and parallel implementation of encoding by RS erasure code.

Firstly, configuring the disaster recovery relationship, but all operations of encoding in this relationship must be carried out in a disaster recovery center, and should not select the optimal disaster recovery center via the load balancing module. We use the software named DebugView to count the encoding time of I/O block with different size in the disaster recovery center 1. The results are shown in Table 5 and Figure 6.

Table 5. Performance Comparison

I/O block size(KB)	(2,4) serial encoding time(s)	(2,4) parallel encoding time(s)	(3,6) serial encoding time(s)	(3,6) parallel encoding time(s)
2	0.000294	0.000227	0.000456	0.000321
4	0.000522	0.0004	0.000898	0.000681
8	0.001618	0.000657	0.00221	0.000889
16	0.004043	0.001588	0.006154	0.00342
32	0.016	0.004708	0.022458	0.006269
64	0.037234	0.018180	0.052125	0.02698
128	0.06726	0.023213	0.10326	0.050516
256	0.26809	0.069695	0.369852	0.15159
512	0.573648	0.226845	0.71336	0.349685
1024	0.926897	0.45836	1.43251	0.695123

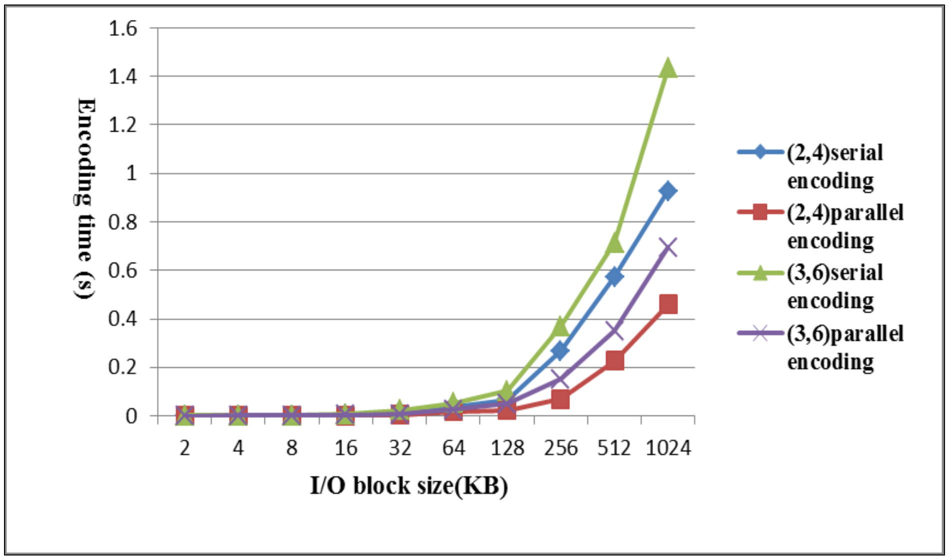

Figure 6. Performance Comparison

As can be seen from the above results, parallel encoding gets better performance than serial encoding. Taking (2, 4) mode as an example, the average performance of the parallel encoding improves on 53% compared with serial encoding. Meanwhile, with the increase of the number of disaster recovery centers in the disaster recovery relationship, the encoding time also increases.

6. Conclusions

This paper designs and implements a distributed disaster recovery system based on PRS algorithm. The system, called OsnDDR, can ensure the source data security in the case of multi-node damage through 1+1+N distributed architecture. Meanwhile, this paper also proposes a load balancing strategy suitable for OsnDDR, which enables the load balancing of each disaster recovery center. In addition, the RS erasure code is achieved by the parallel way to improve the system performance. Experimental results show that OsnDDR can guarantee the normal operation of users' applications, implement multipoint disaster tolerance of users' source data and ensure source data security under the conditions of multi-node damage.

Acknowledgement

This paper is granted by National Natural Science Foundation of China under Grant(No. 61202094), Zhejiang Provincial Natural Science Foundation(No.LY13F020045, LY13F020047), China Postdoctoral Science Foundation(No. 2013M541780) and The National Key Technology R&D Program under Grant(No.2012BAH24B04).

References

[1] S. Ghemawat, H. Gobioff and S. T. Leung, "The Google file system", ACM SIGOPS Operating Systems Review, **(2003)** October, pp. 29-43.
[2] S. A. Weil, "Ceph: Reliable, scalable, and high-performance distributed storage", University of California **(2007)**.
[3] N. Xiao, J. W. Shu, F. Liu and M. Q. Li, "Research on the state of art of storage technologies and future trend", The Annual Report in 2008 on the Development of Computer Science and Technology, **(2009)**, pp. 12-48.

[4] C. M. Lawler, M. A. Harper, S. A. Szygenda and M. A. Thornton, "Components of disaster-tolerant computing: analysis of disaster recovery", IT application downtime and executive visibility, International Journal of Business Information Systems, vol. 3, no. 3, **(2008)**, pp. 317-331.
[5] Symantec Corporation, Veritas Volume Replicator Option by Symantec-A Guide to Understanding Volume Replicator. Engineering White Paper, **(2006)**.
[6] EMC Corporation. EMC SRDF: Zero data loss solution for extended distance replication. Technical Report, **(2009)**.
[7] H. Patterson, S. Manley, M. Federwisch, D. Hitz, S. Kleiman and S. Owara, "SnapMirror: file system based asynchronous mirroring for disaster recovery", Proceedings of the 1st USENIX Conference on File and Storage Technologies, **(2002)** January, pp. 9-9.
[8] M. Flouris and A. Bilas, "Clotho: Transparent Data Versioning at the Block I/O Level", MSST, **(2004)**, pp. 315-328.
[9] D. Hitz, J. Lau and M. A. Malcol, "File System Design for an NFS File Server Appliance", USENIX winter, **(1994)**, pp. 94.
[10] J. Shu, B. Li and W. Zheng, "Design and implementation of an SAN system based on the fiber channel protocol", Computers, IEEE Transactions, vol. 54, no. 4, **(2005)**, pp. 439-448.
[11] Q. Xia, L. Jia, Q. Shen and D. Feng, "Research of integrating military storage resources based on storage virtualization", Networking, Architecture, and Storages 2006. IWNAS'06. International Workshop on. IEEE, **(2006)** August, pp. 2.
[12] I. Foster, Y. Zhao, I. Raicu and S. Lu, "Cloud computing and grid computing 360-degree compared", Grid Computing Environments Workshop, **(2008)** November, pp. 1-10.
[13] Internet Small Computer Systems Interface (iSCSI). http://tools.ietf.org/html/rfc3720.
[14] T. Wood, E. Cecchet, K. K. Ramakrishnan, P. Shenoy and A. Venkataramani, "Disaster recovery as a cloud service: Economic benefits & deployment challenges", 2nd USENIX Workshop on Hot Topics in Cloud Computing, **(2010)** June.
[15] A. Baker, "The Windows 2000 device driver book: a guide for programmers", Prentice Hall Professional, **(2001)**.
[16] P. Orwick and G. Smith, "Developing drivers with the windows driver foundation", O'Reilly Media, Inc, **(2010)**.
[17] G. Zhu, Z. Liangchen and L. Guonian, "The access control technology of spatial data files based on file system filter driver", Communication Technology, ICCT 2008. 11th IEEE International Conference, **(2008)** November, pp. 734-737.
[18] IBM Corporation. A Disaster Recovery very Solution Selection Methodology, **(2004)** February.
[19] D. Roman, U. Keller, H. Lausen, J. Debruijn, R. Lara., M. Stollberg and D. Fensel, "Web service modeling ontology", Applied ontology, vol. 1, no. 1, **(2005)**, pp. 77-106.
[20] Y. Li, Y. Liu, L. Zhang, G. Li, B. Xie and J. Sun, "An exploratory study of web services on the internet", Web Services, ICWS 2007, IEEE International Conference, **(2007)** July, pp. 380-387.
[21] P. Alpatov, G. S. Baker, H. C. Edwards, J. A. Gunnels, G. Morrow, J. Overfelt and Y. J. J. Wu, "PLAPACK: Parallel Linear Algebra Package", PPSC, **(1997)** March.
[22] R. C. Agarwal, S. M. Balle, F. G. Gustavson, M. Joshi and P. Palkar, "A three-dimensional approach to parallel matrix multiplication", IBM Journal of Research and Development, vol. 39, no. 5, **(1995)**, pp. 575-582.
[23] J. Menon, D. A. Pease, R. Rees, L. Duyanovich and B. Hillsberg, "IBM Storage Tank—a heterogeneous scalable SAN file system", IBM Systems Journal, vol. 42, no. 2, **(2003)**, pp. 250-267.
[24] L. P. Slothouber, "A model of web server performance", Proceedings of the 5th International World wide web Conference, **(1996)** May.
[25] R. Wood, "Future hard disk drive systems", Journal of magnetism and magnetic materials, vol. 321, no. 6, **(2009)**, pp. 555-561.
[26] N. Allen, "Don't waste your storage dollars: what you need to know", Research note, Gartner Group, **(2001)**.
[27] L. Rizzo, "Effective erasure codes for reliable computer communication protocols", ACM SIGCOMM Computer Communication Review, vol. 27, no. 2, **(1997)**, pp. 24-36.
[28] A. Krioukov, L. N. Bairavasundaram, G. R. Goodson, K. Srinivasan, R. Thelen, A. C. Arpaci-Dusseau and R. H. Arpaci-Dusseau, "Parity Lost and Parity Regained", FAST, vol. 8, **(2008)** February, pp. 127-141.
[29] J. S. Plank, J. Luo, C. D. Schuman, L. Xu and Z. A. Wilcox-O'Hearn, "Performance Evaluation and Examination of Open-Source Erasure Coding Libraries for Storage", FAST, vol. 9, **(2009)** February, pp. 253-265.
[30] O. Khan, R. Burns, J. Plank and C. Huang, "In search of I/O-optimal recovery from disk failures", Proceedings of the 3rd USENIX conference on Hot topics in storage and file systems, **(2011)** June, pp. 6-6.
[31] Iometer. http://www.iometer.org/.

Authors

Yongjian Ren, he received the PhD degree in Computing Application Technology from Florida Atlantic University, in 1998. He is currently a professor in software engineering in Hangzhou Dianzi University, China. His research interests include Cloud Computing and Mass Storage.

Huijia Xuan, he received the Bachelor Degree of Computer Science and Technology in Hangzhou Dianzi University, Zhejiang, China, in 2011. He is now studying the Master of Computer System Structure in Hangzhou Dianzi University, China. His research interest is Network Storage Technology.

Jilin Zhang, he received the PhD degree in Computer Application Technology from University of Science & Technology Beijing, Beijing, China, in 2009. He is currently a lecture in software engineering in Hangzhou Dianzi University, China. His research interests include High Performance Computing and Cloud Computing.