Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Extending attributes page: a scheme for enhancing the reliability of storage system metadata*

Juan WANG†, Dan FENG, Fang WANG, Cheng-tao LU

(*MOE Key Lab of Data Storage System, Huazhong University of Science and Technology, Wuhan 430074, China*)

†E-mail: rabbitmam@126.com

Received June 25, 2008; Revision accepted Oct. 6, 2008; Crosschecked May 27, 2009

Abstract: In an object-based storage system, a novel scheme named EAP (extending attributes page) is presented to enhance the metadata reliability of the system by adding the user object file information attributes page for each user object and storing the file-related attributes of each user object in object-based storage devices. The EAP scheme requires no additional hardware equipments compared to a general method which uses backup metadata servers to improve the metadata reliability. Leveraging a Markov chain, this paper compares the metadata reliability of the system using the EAP scheme with that using only metadata servers to offer the file metadata service. Our results demonstrate that the EAP scheme can dramatically enhance the reliability of storage system metadata.

Key words: Metadata server, Metadata reliability, Markov model, Extending attributes page (EAP)
doi:10.1631/jzus.A0820485 **Document code:** A **CLC number:** TP31

INTRODUCTION

With the development of network storage technology, the object-based storage device (OSD) protocol (Mesnier *et al.*, 2003; 2005; Lohmeyer and Evans, 2008) is considered as the new standard of next generation network storage systems and becomes noticeable in the storage domain. Different from traditional block-based [e.g., storage area network (Hulen *et al*., 2002; Clark, 2003)] or file-based [e.g., network-attached storage (Nagle *et al.*, 1999; Gibson and Meter, 2000)] storage systems, object-based storage systems (OBSs) employ an expressive storage interface—the object interface.

In an OBS, end-user data such as files are eventually mapped into user objects, which are stored in OSDs and identified by an immutable, globally unique 128-bit identifier. A file is still accessed by applications using its file name, and the mapping of files to objects is implemented by a metadata server (MDS) which records the mapping relationships between files and objects.

In existing traditional large-scale storage systems, including storage area networks (SANs) and OBSs (Abd-El-Malek *et al.*, 2005; Weil *et al.*, 2006; Braam, 2007), MDSs are the bridge between clients and storage devices storing user data. Once MDSs fail because of hardware, software or network faults, storage systems could not holistically afford any storage service even though all the storage devices and data on them remain alive.

Because none but the MDS stores the files and directories information of user objects, the MDS is crucial for the whole storage system. File metadata stored only in a single MDS is easy to become a single point of failure, and hence we need to seek a way to provide highly-reliable metadata services.

Most storage systems avoid a single point of failure of metadata by offering failover metadata services with the use of backup MDSs; that is, file metadata may be stored in several different MDSs as

* Project supported by the National Natural Science Foundation of China (No. 60873028) , the National Basic Research Program (973) of China (No. 2004CB318201), the Program for New Century Excellent Talents in University (No. NCET-04-0693), and the Innovational Group Project (No. IRT0725), China

file metadata replicas. Thus, no less than two file metadata replicas are preserved in generic storage systems (Ghemawat *et al.*, 2003; Tang *et al.*, 2004) to guarantee metadata reliability. For OBSs, we propose a scheme named EAP (extending attributes page), which adds a user object file information attributes page (OFAP) for each user object and stores the file-related attributes of each user object in OSDs to enhance the metadata reliability of OBSs. When all MDSs fail, the metadata of the storage system can still be accessed by the OFAP as long as each user object is available. Therefore, the EAP scheme enhances the metadata reliability of OBSs.

DESIGN OVERVIEW

The OSD object abstraction is designed to assign the responsibilities of storage space management to storage devices (Lohmeyer and Evans, 2008), and the MDS takes charge of the global namespace management of a storage system.

OBS workflow

An OBS consists of a single MDS or MDS cluster, multiple OSDs and multiple clients. A file, accessed by applications using its file name, may be divided into one or multiple user objects identified by an immutable, globally unique 128-bit identifier (object ID), which is made up of a partition identifier and a user object identifier assigned by the MDS at the creation time of the user object. OSDs store user objects and their attributes on local disks or RAIDs (redundant array of inexpensive disks).

An MDS maintains storage system metadata, which include the directory hierarchy metadata, the mapping metadata from files to user objects, and the current location metadata of user objects. When a client needs to access a file, the client firstly interacts with the MDS to ask which OSDs it should contact to acquire user objects. The MDS converts a file operation to one or several object operations and informs the client of the OSDs' IP addresses (OSD_IP) and the corresponding object IDs of the requested file. Then the client interacts directly with the corresponding OSDs for the subsequent objects read or write operations. Fig.1 shows the workflow.

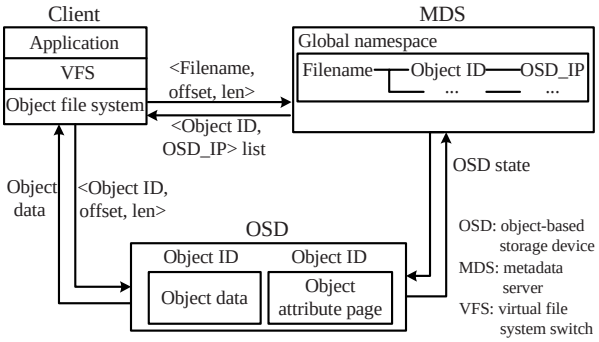

Fig.1 Workflow in an object-based storage system

The mapping process of files to objects is accomplished by the MDS, which records the mapping relationship between files and objects. In current OBSs, a user object is identified only by the object ID, blind to the related files and directories information. Hence, once MDSs fail because of hardware, software or network faults, storage systems could not holistically afford any storage service even though all OSDs and user object data stored on them are intact and correct.

User object attribute

User object attributes, which describe the specific characteristics of a user object, are organized into many attributes pages, and each attributes page consists of some attribute items with similar sources or uses. An attribute item is indexed by a pair (attributes page number, attribute number). Applications can define some attributes that may reflect their requirements by extending an attributes page in the OSD SCSI (small computer system interface) protocol (Lohmeyer and Evans, 2008).

Since only the MDS knows the mapping relationship between files and objects, the MDS is crucial and easy to become a single point of failure, and thus we need to seek a way to provide highly-reliable metadata services.

Most storage systems eliminate the single point of failure of metadata by offering failover metadata services, which improves the metadata reliability of storage systems. In an OBS we can enhance the metadata reliability by virtue of extending user object attributes.

The EAP scheme proposed in this work is an extension for the current OSD specification, expressing the hints of files and directories information related to user objects.

EAP scheme

The user OFAP is added for each user object in the EAP scheme. The page defines fundamental file information attributes related to a user object.

The attribute items defined in the OFAP (Table 1) are described as follows.

The fragment identifier field indicates to which fragment of the file a user object belongs. For example, suppose that a file '/root/1.avi' is sequentially mapped into three user objects: O1, O2 and O3, and stored on OSD1, OSD2 and OSD3, respectively. Consequently, the fragment identifier in the OFAP of user object O2 is 2, because O2 belongs to the second fragment of the file.

The object sum field specifies how many objects a file is mapped into. For the above mentioned example, the object sum is 3.

The field about the mapping strategy of files to objects specifies how a file is mapped into one or multiple objects. Reversibly, we can know how to reconstruct a file from related objects using the attribute. For the example mentioned above, the mapping strategy of files to objects is the sequential stripping pattern.

The first object ID field represents the object identifier of the first fragment object of a file. As the above example illustrates, the first object ID in the OFAP of O2 is the object ID of O1.

The first OSD IP field describes the IP address of the OSD on which the first object of a file is stored. In the above example, the first OSD IP in the OFAP of O2 is the IP address of OSD1.

The filename length field indicates the number of bytes of the following filename field. For example, the filename length of file '/root/1.avi' is 11 bytes.

The filename field describes the name of the file that a user object belongs to. As regards the above example, the filename in the OFAP of the user object O2 is '/root/1.avi'. The field length is variable, and is indicated by the foregoing filename length field because different filenames occupy different numbers of bytes, and some storage space will be wasted if the filename length is fixed. From a metadata study (Agrawal *et al.*, 2007), we find that files deeper in the namespace tree tend to be orders-of-magnitude smaller than shallower files. This illuminates indirectly that the number of files having long filenames is small, and it is more valuable to store each filename at a variable length.

Table 1 gives the necessary fields of the OFAP, and some selective fields can also be defined. For example, a file type field can be used in a storage symbolic link in addition to the common file and directory. In this way, the filename of a linked file is stored in the user object data of the symbolic link, and the file type in the OFAP of the user object is the symbolic link.

When the MDS of a storage system fails, the directory hierarchy metadata, the mapping metadata from files to user objects, and the current location metadata of these user objects can still be accessed or rebuilt through the OFAP, which enhances the reliability of storage system metadata. The process of reconstructing the namespace is complex, and read-only is permitted during the recovery to ensure the strong consistency of namespace.

Table 1 User object file information attributes page

Attribute number (h)	Length (byte)	Attribute	Client set table	OSD logical unit
0	40	Page identification	No	Yes
1	2	Fragment identifier	Yes	No
2	2	Object sum	Yes	No
3	2	Mappings strategy of files to objects	Yes	No
4	8	The first object ID	Yes	No
5	4	The first OSD IP	Yes	No
...	...	...	...	...
2+2×n	8	The nth object ID	Yes	No
3+2×n	4	The nth OSD IP	Yes	No
4+2×n	2	Filename length	Yes	No
5+2×n	Variable	Filename	Yes	No
(6+2×n)~FFFFFFF		Reserved		

Some people believe that the EAP scheme is expensive from the perspectives of both storage capacity and synchronization overhead when a large file is involved. However, this is not the case. The synchronization complexity of maintaining the extended attributes page is small, because the OFAP of a user object is modified by the set attributes command only when another new user object of a file is created or another existing object of a file is deleted, whereas the majority of object operations are the read or write operations in a real system. The storage space occupied by the OFAP of each user object is small compared with that of data and many attributes pages of each user object defined in the current SNIA (storage networking industry association) OSD specification (Lohmeyer and Evans, 2008).

METADATA RELIABILITY MODEL ANALYSIS

Since both the mean time to failure (MTTF) and the mean time to repair (MTTR) of a single machine follow the exponential distribution, the continuous-time Markov chain can be leveraged to analyze the reliability of storage system metadata.

Mean-time-to-metadata-loss (MTTML) of a storage system is calculated and analyzed using different methods. In the following subsections, the method using only MDSs to offer file metadata services is called the general method, and file metadata replicas can only be obtained from MDSs. Once some MDSs storing all metadata replicas of a file fail, the user objects of the file could not be accessed even though these objects remain alive in OSDs. The method using the EAP scheme is called the improved method, in which the storage service is still available even when all MDSs fail.

Markov model of the general method

Generally a storage system employs P ($P\geq 2$) different MDSs to store P replicas of file metadata to ensure highly-reliable metadata services. One MDS acts as the primary MDS of file metadata to deal with metadata requests. Meanwhile, the remaining $P-1$ MDSs are used as the backup MDSs of the file metadata to offer the failover metadata service and as recovery sources when the primary MDS fails.

We assume that, for a file metadata replica stored in MDSs, the MTTF is $MTTF_1$ and the MTTR is $MTTR_1$. The failure rate λ_1 and the recovery rate μ_1 of a file metadata replica stored in MDSs are equal to $1/MTTF_1$ and $1/MTTR_1$, respectively.

When all P MDSs storing all replicas of file metadata fail, none of the user objects of the file could be accessed even though the objects are still intact and correct.

There are $P+1$ states for the Markov model of the general method shown in Fig.2. The symbol P is the total number of the replicas of file metadata. State P is the initial state with P available replicas. State $P-1$ indicates that $P-1$ replicas are available and a replica fails. State 1 denotes that only a replica is available, and state 0 means that all replicas are lost and MDSs cannot supply metadata services any more. The file metadata state transits from one state to another when a replica failure event or a replica recovery event is induced by an MDS failure or recovery event. In state P, the P replicas would potentially fail, so the transitional probability from state P to state $P-1$ is $P\lambda_1$. In state $P-1$, the transitional probability from state $P-1$ to state P is μ_1 as the recovery rate of the failed replica is μ_1, and the transitional probability from state $P-1$ to state $P-2$ is $(P-1)\lambda_1$ as the left $P-1$ replicas would potentially fail. In state 1, the transitional probability from state 1 to state 0 is λ_1 because the left unique replica would potentially fail, and state 0 cannot transit to state 1 because no available replicas can be used as the recovery source. The states can be divided into two categories: the normal service and the failure service. The former category ($\{P, P-1, \ldots, 1\}$ in Fig.2) is the set of the states that can support the normal service of metadata while the latter ($\{0\}$ in Fig.2) is the set of the states that cannot provide metadata services.

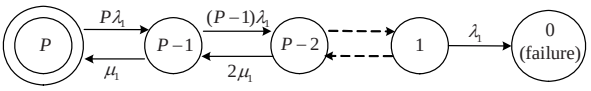

Fig.2 Markov model of the general method

Given the probabilities of states P, $P-1$, …, 0, i.e., $P_P(t)$, $P_{P-1}(t)$, …, $P_0(t)$ respectively, the MTTML of the general method, $MTTML_1$, can be expressed by the probabilities of these states in the normal service category:

$$MTTML_1=\int_0^{\infty}(P_P(t)+P_{P-1}(t)+\cdots+P_1(t))\mathrm{d}t. \quad (1)$$

$P_i(t)$ (i=P, P−1, …, 1) can be derived from the following differential equation (Ross, 1983; Billinton and Allan, 1992):

$$[P_P'(t), P_{P-1}'(t), ..., P_0'(t)] = [P_P(t), P_{P-1}(t), ..., P_0(t)]\boldsymbol{Q}_1. \tag{2}$$

The matrix $\boldsymbol{Q}_1$ can be obtained from Fig.2, as given in Eq.(3) at the bottom of this page.

The element q_{ij} ($0 \le i, j \le P$) of $\boldsymbol{Q}_1$ can be described as follows:

$$q_{ij} = \begin{cases} i\mu_1, & j = i-1,\ 1 \le i \le P-1, \\ -i\mu_1 - (P-i)\lambda_1, & j = i,\ 0 \le i \le P-1, \\ (P-i)\lambda_1, & j = i+1,\ 0 \le i \le P-1, \\ 0, & \text{otherwise.} \end{cases}$$

Eq.(2) can be solved by Eq.(3) and the initial state values $P_P(0)=1$, $P_{P-1}(0)=\ldots= P_0(0)=0$ when the value P is known. Finally, $MTTML_1$ can be figured out by Eq.(1).

Markov model of the improved method

We assume that P different MDSs are employed to support P replicas of file metadata, and OFAPs of objects are stored in Q different OSDs when a file is mapped into Q objects. When all P MDSs fail, a file can still be accessed as long as the OSDs storing all user objects of the file remain alive.

We also assume that the failure rate is λ_1 and the recovery rate is μ_1 for a file metadata replica stored in MDSs, while the failure rate is λ_2 and the recovery rate is μ_2 for an OFAP stored in OSDs.

There are (P+1)(Q+1) states for the Markov model of the improved method (Fig.3), each being depicted by a tuple <x, y> ($0 \le x \le P$, $0 \le y \le Q$). Therein the symbol x indicates that x replicas of file metadata are still available in MDSs and the symbol y represents that there are still y OFAPs in OSDs.

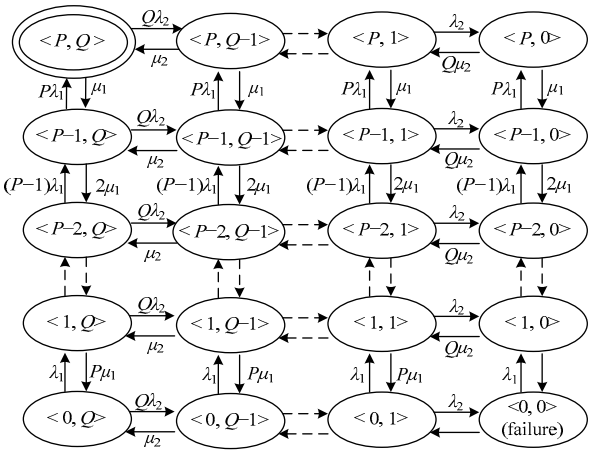

Fig.3 Markov model of the improved method

State <P, Q> represents the initial state with all P file metadata replicas stored in MDSs and all Q OFAPs stored in OSDs available. State <0, 0> denotes that all file metadata replicas stored in MDSs and OFAPs stored in OSDs are unavailable. The file metadata state transits from one state to another when either of such events as a file metadata replica failure or recovery event, or an OFAP failure or recovery event is induced by an MDS or OSD failure or recovery event. In state <x, y>, the transitional probability from state <x, y> to state <x−1, y> is $x\lambda_1$ because the left x replicas would potentially fail, and the transitional probability from state <x, y> to state <x, y−1> is $y\lambda_2$ because the left y OFAPs would potentially fail. In state <x, y>, the transitional probability from state <x, y> to state <x+1, y> is $(P-x)\mu_1$ because the failed $P-x$ replicas would be potentially recovered, and the transitional probability from state <x, y> to state <x, y+1> is $(Q-y)\mu_2$ because the failed $Q-y$ OFAPs would be potentially recovered. State <0, 0> cannot transit to any other state because none of the available replicas or OFAPs can be used as a recovery source.

Using the same method, the MTTML of the improved method, $MTTML_2$, can be calculated.

$$\boldsymbol{Q}_1 = \begin{bmatrix} -P\lambda_1 & P\lambda_1 & 0 & \ldots & \ldots & \ldots & 0 \\ \mu_1 & -\mu_1 - (P-1)\lambda_1 & (P-1)\lambda_1 & \ldots & \ldots & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & 0 \\ 0 & 0 & 0 & \ldots & (P-1)\mu_1 & -(P-1)\mu_1 - \lambda_1 & \lambda_1 \\ 0 & 0 & 0 & \ldots & 0 & 0 & 0 \end{bmatrix}. \tag{3}$$

Discussion of MTTML

The $MTTML_1$ and $MTTML_2$ for a few small values of P and Q are recorded in Tables 2 and 3, respectively. Larger values of P and Q are not listed due to expression complexity and space limitation.

Table 2 $MTTML_1$ of the general method for some small values of P

P	$MTTML_1$
1	$1/\lambda_1$
2	$(3\lambda_1+\mu_1)/(2\lambda_1^2)$
3	$(11\lambda_1^2+7\lambda_1\mu_1+2\mu_1^2)/(6\lambda_1^3)$

Table 3 $MTTML_2$ of the improved method for some small values of P and Q

P	$MTTML_2$	
	Q=0	Q=1
0	–	$1/\lambda_2$
1	$1/\lambda_1$	$\frac{1}{\lambda_1}+\frac{(\lambda_1+\mu_1)(\lambda_1+\mu_2)}{\lambda_1\lambda_2(\lambda_1+\lambda_2+\mu_1+\mu_2)}$

The state transitions with the Markov model of the improved method are more complicated than those of the general method. Actually, we can find that the Markov model of the general method is a special case of that for the improved method when Q is equal to zero.

Using the *MTTML* computed from the Markov models, we can compare the *MTTML* of the different methods, as shown in Fig.4. For each configuration, the *MTTML* for $MTTR_1$=$MTTR_2$=4 h varies with $MTTF_1$=$MTTF_2$=*MTTF*. As the results indicate, the *MTTML* of the improved method is significantly larger than that of the general method, increasing as Q value grows when the value P is fixed.

Fig.4a indicates that the *MTTML* is small when there is only one file metadata replica in the general method. The *MTTML* can be largely increased when some file metadata replicas are stored in different MDSs or the EAP scheme is employed alternatively.

In a real system, although the *MTTML* increases as P or Q increases, the values of P and Q are required to be not too large because the overhead of maintaining file metadata consistency is raised when the number of file metadata replicas stored in different MDSs increases or the number of OFAPs increases when the EAP scheme is employed. It can be concluded from Figs.4a and 4b that P=2 and Q=1 or P=1 and Q=2 is already enough to keep the *MTTML* large.

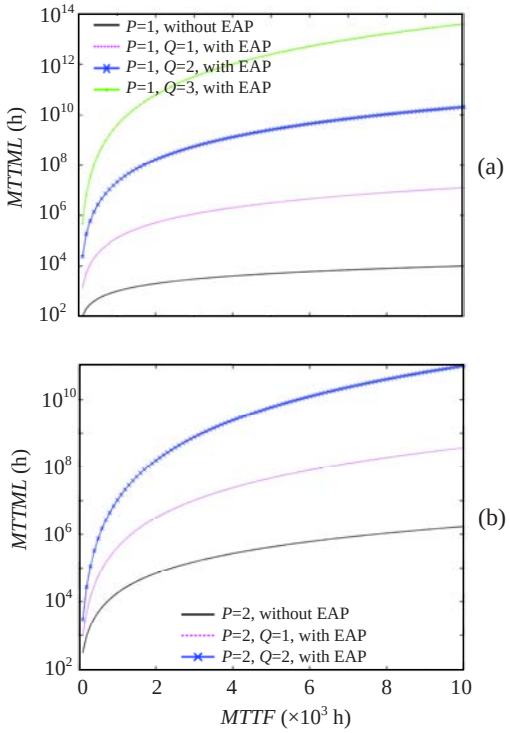

Fig.4 *MTTML* of different configurations varied with *MTTF* for $MTTR_1$=$MTTR_2$=4 h. (a) A single MDS; (b) Two MDSs

According to the above analysis, only the first two of the n objects of a file include OFAPs when the file is mapped into n (n≥2) user objects.

In Figs.4a and 4b, $MTTR_1$ and $MTTR_2$ are assumed to be equal. In fact, $MTTR_2$ can be smaller than $MTTR_1$ as a result of file metadata centrally stored in MDSs and OFAPs decentralizedly stored on different OSDs. In consequence, there are more recovery sources and recovery targets to accelerate the process of metadata recovery in the EAP scheme. Fig.5 illustrates that the *MTTML* of the different methods varies with the rate of $MTTR_2/MTTR_1$ in which $MTTR_1$=4 h and $MTTF_1$=$MTTF_2$=5×10^4 h. It shows that the *MTTML* with P=2 in the general method is the same as that with P=1 and Q=1 in the improved method when $MTTR_1$ is equal to $MTTR_2$, and that the *MTTML* of the improved method is generally larger than that of the general method since $MTTR_2$ is smaller than $MTTR_1$.

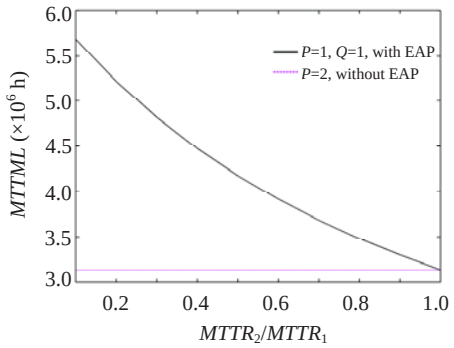

Fig.5 *MTTML* of different configurations varied with $MTTR_2/MTTR_1$ for $MTTR_1$=4 h

We can conclude that thanks to its need of fewer MDSs (i.e., lower hardware requirements) to gain the approximate *MTTML*, the improved method is better than the general method.

In the common circumstances of storage systems, the actual number of user object replicas is not less than 2, which dramatically improves the reliability of user objects, and the metadata reliability is also greatly enhanced in virtue of the EAP scheme. When the EAP scheme is not adopted, the metadata reliability depends on the sum of the dedicated MDSs in which multiple file metadata replicas are stored. In this situation, the metadata reliability remains a Gordian knot albeit there is a rise in the number of user object replicas.

CONCLUSION

This paper presents the EAP scheme to enhance metadata reliability, which requires no additional hardware equipments, but takes full advantage of the expressive object interface affiliated with an OBS via adding the user OFAP. In the EAP scheme, the adoption of the user OFAP temporarily provides failover metadata services when MDSs fail, which significantly enhances the reliability of storage system metadata.

We develop a simple Markov model for metadata by using only MDSs to offer file metadata services and then propose another Markov model for metadata utilizing the EAP scheme. Furthermore, from our comparison of the MTTMLs in the two different methods, we find that the metadata reliability of storage systems employing the EAP scheme is better than that of the general method.

In summary, the reliability of storage system metadata can be dramatically enhanced by adding the user OFAP. Furthermore, the EAP scheme does not exclude other schemes of improving the reliability of storage system metadata, and hence can be a good supplement for achieving higher metadata reliability.

References

Abd-El-Malek, M., Courtright, W.V., Cranor, C., Ganger, G.R., Hendricks, J., Klosterman, A.J., Mesnier, M., Prasad, M., Salmon, B., Sambasivan, R.R., *et al*., 2005. Ursa Minor: Versatile Cluster-based Storage. Proc. 4th USENIX Conf. on File and Storage Technology, p.59-72.

Agrawal, N., Bolosky, W.J., Douceur, J.R., Lorch, J.R., 2007. A Five-year Study of File-system Metadata. Proc. 5th USENIX Conf. on File and Storage Technologies, p.31-45.

Billinton, R., Allan, R.N., 1992. Reliability Evaluation of Engineering System: Concepts and Techniques. Plenum Press, New York.

Braam, P.J., 2007. Lustre File System: High-performance Storage Architecture and Scalable Cluster File System. Whiter Paper, Sun Microsystems, Inc., Santa Clara, CA, USA.

Clark, T., 2003. Designing Storage Area Networks: A Practical Reference for Implementing Storage Area Networks. Addison-Wesley Longman Publishing Co., Inc., USA.

Ghemawat, S., Gobioff, H., Leung, S.T., 2003. The Google File System. Proc. 19th ACM Symp. on Operating Systems Principles, p.29-43. [doi:10.1145/945445.945450]

Gibson, G.A., Meter, R.V., 2000. Network attached storage architecture. *Commun. ACM*, **43**(11):37-45. [doi:10.1145/353360.353362]

Hulen, H., Graf, O., Fitzgerald, K., Watson, R.W., 2002. Storage Area Networks and the High Performance Storage System. Proc. 19th IEEE/10th NASA Goddard Conf. on Mass Storage Systems and Technologies, p.225-240.

Lohmeyer, J.B., Evans, M., 2008. Information Technology-SCSI Object-based Storage Device Commands-2 (OSD-2). INCITS T10 Working Draft. Available from: http://www.t10.org/ftp/t10/drafts/osd2/osd2r04.pdf

Mesnier, M., Ganger, G.R., Riedel, E., 2003. Object-based storage. *IEEE Commun. Mag.*, **41**(8):84-90. [doi:10.1109/MCOM.2003.1222722]

Mesnier, M., Ganger, G.R., Riedel, E., 2005. Object-based storage: pushing more functionality into storage. *IEEE Potent.*, **24**(2):31-34. [doi:10.1109/MP.2005.1462464]

Nagle, D.F., Ganger, G.R., Butler, J., Goodson, G., Sabol, C., 1999. Network Support for Network-attached Storage.

Proc. Hot Interconnects, p.45-50.

Ross, S., 1983. Stochastic Processes. John Wiley Press, New York, USA.

Tang, H., Gulbeden, A., Zhou, J.Y., Strathearn, W., Yang, T., Chu, L.K., 2004. The Panasas ActiveScale Storage Cluster-delivering Scalable High Bandwidth Storage. Proc. ACM/IEEE SC Conf., p.53-62. [doi:10.1109/SC.2004.57]

Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D.E., Maltzahn, C., 2006. Ceph: A Scalable, High-performance Distributed File System. Proc. 7th Symp. on Operating Systems Design and Implementation, p.307-320.