IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 6, Ver. V (Nov.-Dec. 2016), PP 104-112
www.iosrjournals.org

Applicability Erasure Coding on Non-Traditional Cloud Storage Services for Storage Fault Tolerance

L. Srinivasa Rao[1], Dr.I. Raviprakash Reddy[2]
[1]*Mother Teresa Institute of Science and Technology Sathupally, Telangana State, India*
[2] *G Narayanama Institute of Technology and Science, Hyderabad, Telangana State, India*

Abstract: *With the fastest growing nature of application migrating to cloud, the problem of choosing the best suitable cloud service for storage is always a challenge. The cloud storage services are majorly paid service and some of them are designed to deal best with specific kinds of data like read only or update only. Some of the services are designed to match small amount of data blocks and some of the services are designed to best match the larger blocks of data. Many of the application uses file storage, rather than RDBMS storage structure on cloud services. Moreover the failure is an added component of risk to cloud storage. Though replicating data over cloud storage service providers is a common task considering the low cost data recovery. However over replication of data may lead to integrity problem with un-effective cost factor. Multiple works are been addressing the same issue over a period of time to find the most effective replication algorithm. But with a specific focus on domain dependent data and service providers. Hence in this work we propose a comparative study of Erasure algorithm on various cloud service providers. This work also demonstrates a theoretical framework for cost effective storage replication and discussion the performance*.

Keywords: *Cloud Storage, Performance Comparison, Evolution Application, Response Time, Comparison, Erasure, RAID, RAID 4, RAID 5, Array Code, Reed – Solomon Code, Azure, Amazon S3, Performance Matrix.*

I. Introduction

Files in the cloud storage services are usually stored in the container offered by third-party companies. Instead being provided by a single host, the containers are integrated and distributed through centralized management. A great set of works are been conducted on the area of cloud storage services, however the comparative study for storage and retrieval on the loaded network needs to be studies in real time. Cloud computing services can be seen as either computing or storage offering [1] [2] [3] [4].

The storage systems on cloud came a long way in terms of capacity and latency time improvement. All the storage hardware types are commonly failing to protect data during failures and unable to restrict data loss. The type of failure can be not having control on getting disk sectors corrupted or the entire disk is becoming unusable. The storage services have some self-protecting mechanism as extra-corrective information that can detect changing of few bits from the original data and can still retrieve the originally stored data. However there are situations when multiple bits change unexpectedly, then the self-protecting mechanism detects that as hardware failure and storage devices become un-usable. This situations lead to loss of data [5] [6].

To handle these types of anomalies, the storage systems depend on Erasure codes. The Erasure code deploys the mechanism of assured redundancy to overcome the failures. The most generalized way of implementing this mechanism is replication of data over multiple locations. The most popular and simplest is Redundant Array of Independent Disks or RAID. In that the most basic version of these implementations is RAID – 1, where every data byte is stored in at least two parallel disks. This way the failure may not lead to loss of data as long as a replicated copy of the data is available. This mechanism is easy to achieve, however this leads to many other overhead factors like cost of storage. The storage cost should be at least double than the actual cost. Moreover in any case if both the storage device fails then the complete solution becomes unusable [7] [8]. In the other hand, there are more complex solutions under Erasure methodologies such as well-known Reed-Solomon codes. Reed-Solomon code can overcome high level failures with little less extra storage. These codes provide high level of failure tolerance with reduced cost.

In communication systems the Erasure coding is similar to Error Correcting Codes or ECC. Here the Erasure coding solves the similar types of problems but addresses very different types of problems. In massage communication, the error is caused by changing bits of the data. Here is the different lie between Erasure and message communication as the location of the changing bits is unknown. Hence application of Erasure is restricted [9] [10].

The work here is demonstrated as Section II explores and discuss the proficiency and constraints of multiple storage services with the cost variation based on the data size, in Section III Compare the performance of fault tolerance mechanisms, in the Section IV we demonstrate the architecture of the application for choosing the best suitable storage service automatically, in Section Vwe discuss details of Erasure Code in Section, in the

Section VI we compare the storage services based on the data storage and retrieval inclusive of network and server load parameters along with formulation of these parameters and in Section VII we conclude.

II. Page Layout

As the choice of storage services from cloud is not limited and most of those are configured to give best advantages for specific type of data and operation, we compare most of the services here [11-14]:

A. Dropbox

The Dropbox is a storage service which is available for client side access for Windows systems, Linux Systems, Macintosh systems, Blackberry mobile operating systems, Android mobile operation systems and finally the IPhone operating systems. The free Basic account comes with a paltry 2GB of storage. For document based applications this is huge. The Storage service is good choice for applications using the container for read only data.

Table 1. Cost Comparison for Dropbox.

Data Load	**Cost**
Load in GigaBytes	**Price in US Dollars**
100	**99 USD**
200	**99 USD**
300	**99 USD**
400	**499 USD**
500	**499 USD**
1000	**Not Available**
> 1000	**Not Available**

Here we provide a graphical representation of the cost price comparison:

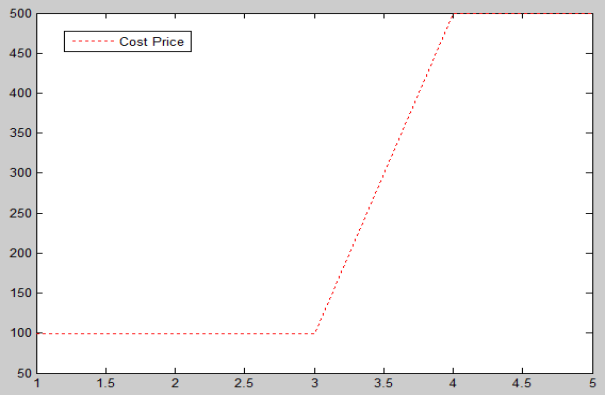

Fig.1. Cost Comparison for Dropbox

Table 2. Support for Mobile Based Cloud Applications in Dropbox

Client OS Type	**Support**
Apple IPhone Operating Systems	Available
Android Mobile Operating Systems	Available
Blackberry Operating Systems	Available
Microsoft Mobile Operating System	Available

B. Google Drive

The most popular cloud storage service is Drive storage from Google. The basic account comes with 15 Giga bytes of storage for a new customer account or an existing account created with Google Email. The highest rated benefit of the Google Drive is the service can be also be integrated with other existing google services for storing various types of data from other services.

Table 3. Cost Comparison for Google Drive

Data Load	**Cost**
Load in Giga Bytes	Price in US Dollars
100	60 USD
200	120 USD
300	120 USD
400	240 USD
500	240 USD
1000	600 USD
> 1000	1200 to 9600 USD

Here we provide a graphical representation of the cost price comparison:

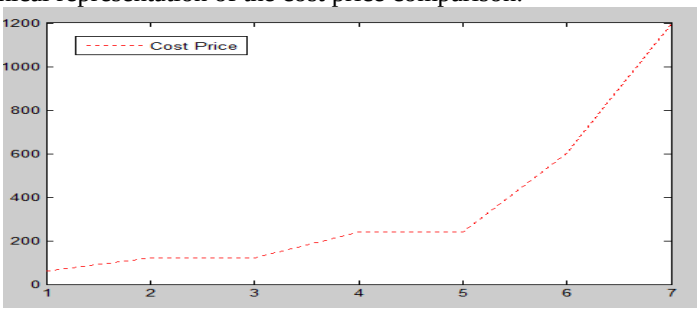

Fig.2. Cost Comparison for Google Drive

Table 4. Support for Mobile Based Cloud Applications in Google Drive

Client OS Type	**Support**
Apple IPhone Operating Systems	Available
Android Mobile Operating Systems	Available
Blackberry Operating Systems	Not Available
Microsoft Mobile Operating System	Not Available

C. Hightail

The previous version of business cloud storage of Hightail was popular by name of YouSendIt. The basic reason for creating the name was the core of the features that Hightail provides. Hightail is majorly known for sharing files, which can be digitally signed for verifications. The core technology behind this provider is link sharing, where the sender can upload a file and the link to that same file can be shared with the recipient. The recipient can click on the link to download the same. This service is popular for business users as it provides the private cloud storage and the desktop version of the client, which can be used for syncing local files to the cloud storage.

Table 5. Cost Comparison for Hightail

Data Load	**Cost**
Load in Giga Bytes	Price in US Dollars
100	Free
200	Free
300	Free
400	Free
500	Free
1000	Free
> 1000	195 USD

Table 6. Support for Mobile Based Cloud Applications in Hightail

Client OS Type	**Support**
Apple IPhone Operating Systems	Available
Android Mobile Operating Systems	Not Available
Blackberry Operating Systems	Not Available
Microsoft Mobile Operating System	Not Available

D. One Drive

The One Drive was previously popular as SkyDrive. The functionalities are mostly same as Dropbox. The most important factor for this storage service is that the client version is available for Windows systems, Linux Systems, Macintosh systems, Blackberry mobile operating systems, Android mobile operation systems and finally the IPhone operating systems. Moreover the supports for social media plug-ins are also available here. This feature makes the application more compatible with other applications to access data directly.

Table 7. Cost Comparison for OneDrive

Data Load	**Cost**
Load in Giga Bytes	Price in US Dollars
100	50 USD
200	100 USD
300	Not Available
400	Not Available
500	Not Available
1000	Not Available
> 1000	Not Available

Here we provide a graphical representation of the cost price comparison:

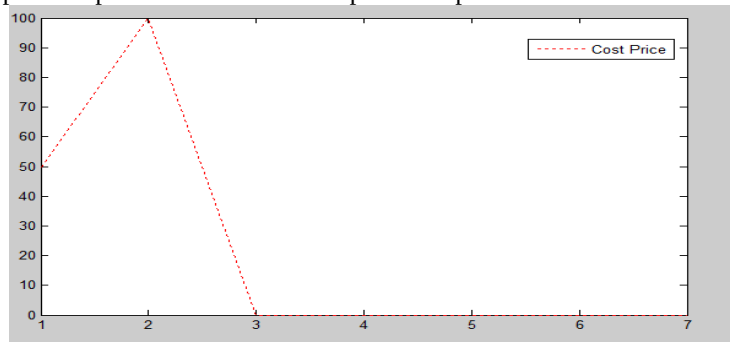

Fig.3. Cost Comparison One Drive

Table 8. Support for Mobile Based Cloud Applications in OneDrive

Client OS Type	**Support**
Apple IPhone Operating Systems	Available
Android Mobile Operating Systems	Available
Blackberry Operating Systems	Available
Microsoft Mobile Operating System	Available

E. One Drive

The SugerSync is majorly popular among business users for its effective and fast online backup solutions. The service can also be used for complete folder and individual file syncing with multiple applications and multiple users. Moreover the service provides a unique function to share the stored content over multiple devices at same point of time but with different permission levels. The most important factor for this storage service is that the client version is available for Android mobile operation systems and also the IPhone operating systems.

Table 9. Cost Comparison for SugerSync

Data Load	**Cost**
Load in Giga Bytes	Price in US Dollars
100	99 USD
200	250 USD
300	250 USD
400	250 USD
500	250 USD
1000	550 USD
> 1000	Pay Per Use

Here we provide a graphical representation of the cost price comparison:

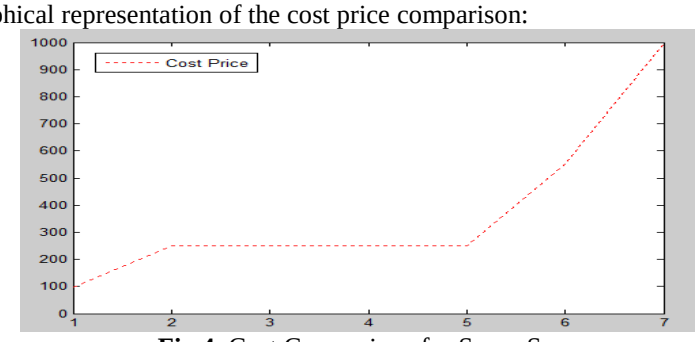

Fig.4. Cost Comparison for Suger Sync

Table 10. Support for Mobile Based Cloud Applications in SugerSync

Client OS Type	**Support**
Apple IPhone Operating Systems	Available
Android Mobile Operating Systems	Available
Blackberry Operating Systems	Available
Microsoft Mobile Operating System	Available

III. Standard Fault Tolerance Mechanisms

The standard fault tolerance mechanism depends on the erasure codes.The basic mechanism can be understood if we assume a collection of n disks are partitioned into k disks. Hence there will be m disks which will hold the coding information as

$$m = n - \sum_{i=1}^{r<n} k_i \quad \text{....Eq 1}$$

Where r denotes number of k multiple of disks

The basic interpretation of the erasure codes can be understood as each disk must hold a z bit word to represent the customer data. If we denote them with d then the total set of codes for k number of disks are considered as

$$z_1, z_2, z_3 .... z_k \quad \text{....Eq 2}$$

Also we consider the codes stored on each every m disk with c, and then the total representation is considered as

$$c_1, c_2, c_3 .... c_k \quad \text{....Eq 3}$$

The coding and the customer data should a linear combination and can be represented as

$$\begin{aligned} c_0 &= a(_{1,0})z_0 + ... + a(_{1,k-1})z_{k-1} \\ c_1 &= a(_{2,0})z_0 + ... + a(_{2,k-1})z_{k-1} \\ &.... \\ &.... \\ c_m &= a(_{m,0})z_0 + ... + a(_{m,k-1})z_{k-1} \end{aligned} \quad \text{....Eq 4}$$

The coefficients “a” are also z bit words. Encoding, therefore,

Simply requires multiplying and adding words, and decodinginvolves solving a set of linear equations with Gaussian elimination or matrix inversion.

Furthermore, we understand the most popular coding techniques here.

A. RAID-4 and RAID-5

The RAID – 4 and RAID – 5 are the simplest form of the erasure codes explained in this work earlier. RAID – 4 and RAID –5 differs from the basic framework as it employs different arrangements of data replication. The framework for RAID – 4 and RAID – 5 are explained here:

The RAID is a modification to MDS code where m=1 and z=1. The basic coding depends on a bit noted as p, where

$$p = z_0 \oplus z_1 \oplus .. \oplus z_{k-1} \quad \text{....Eq 5}$$

In case of any bit changing, the XOR code will identify it for the surviving code.

B. Linux RAID-6

The Linux system RAID – 6 is considered as additional support to RAID – 4 and RAID – 5 as it uses an alternative disk under the framework. This framework proposes an alternation to the MDS as considering the code to be stored in two disks as m=2. Hence the formulation is too simple by using an XOR code:

$$\begin{aligned} p &= z_1 \oplus z_2 \oplus ... \oplus z_k \\ q &= z_1 \oplus 2(z_2) \oplus ... \oplus 2^k(z_k) \end{aligned} \quad \text{....Eq 6}$$

Here the codes called p and q will be stored on alternative disks to ensure the Erasure code to protect the data loss.

C. Array Codes

The framework is called Array code as it is implemented using r X n array of customer data. In this framework the customer data will be stored with the arrangementsas Figure – 2.

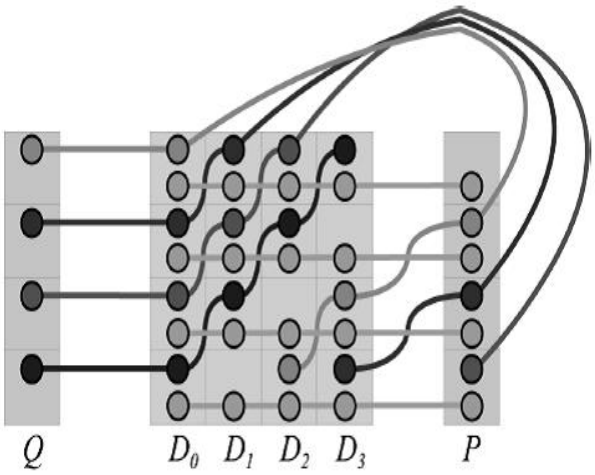

Fig 5: Array Code Storage

The array code with the following parameters: k=4, m=2 (RAID-6), n = k+m = 6, r=4, z=1.

D. Non-MDS Codes

The Non-MDS codes do not allow replication of m storage devices to achieve optimal fault tolerance. The replication of storage devices containing the code is higher than the other frameworks. However the efficiency provided by the Non-MDS codes compared to other frameworks in terms of performance is high. Hence we compare all the types of code frameworks here.

IV. Performance Evaluation Application

The following application is created to demonstrate the load vs response time comparison for the tested cloud service providers.

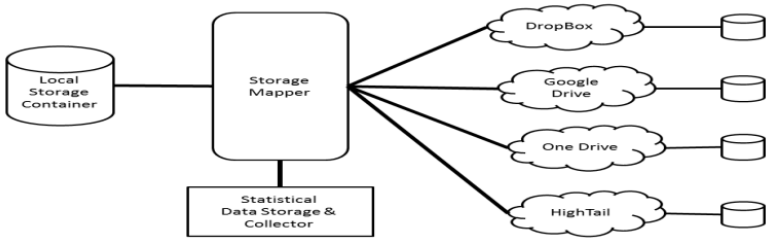

Fig.6. Performance Evaluator Application Architecture

A. **Local Storage Container:** The local storage container stores the load data in form of textual file to be uploaded to multiple cloud storage services for testing. The textual data or the normalized data is considered as the data which cannot be formulized in data base or any other structured data types. However, it is important to understand that the unstructured data can be only text based or rich text including other media data. Most of the time, the unstructured data is collected from multiple applications used for business communication as emails, power point presentations, documents containing images and graphs, collaboration & text sharing documents and finally the instant messengers. The Local Storage is also equipped with a contexter component. The contexter component is basically a normalization component in this application. This component performs a few specific tasks in a specific order like Language detection, Named entity recognition, Anaphoric normalization and Text segmentation. It was assumed that the common text will appear in English and the rest of the process will start with this consideration. The first step is to compute each component block to extract the text by applying the formulas

$$Para_{(x)} = Para \,/\, D(Text_x) \quad \text{…. Eq 7}$$

$$D(Text_x) = \sum_{t=w_1}^{w_n} D_t \quad \text{…. Eq8}$$

$$Para_{(x)_{i,j}} = Text_{(x)_{j,i}} \quad \text{…. Eq 9}$$

$$Para_{(X)_{i,j}} = Text_{(x)_{i+1,i-1}} \quad \text{…. Eq 10}$$

Where, $Para_{(X)}$ is extracted text component

$Para$ is the total text block

$D(Text_x)$ is the domain of recognizable keywords

$Para_{(X)_{i,j}}$ is the extracted text component before mapping

$Text_{(x)_{j,i}}$ is the extracted text component after mapping

The named entry recognition algorithm is to find multiple small normalization-able components of texts (Eq. 1), where the domains of the known keywords are made from each collected keywords (Eq. 2). When the final text is extracted, (which can actually be many pieces of text), the mapping process starts. This mapping process eventually normalizes the unstructured text. The mapping process maps the extracted texts to mapping fields (Eq. 3). Sometimes based on few extracted text, new fields also need to be created (Eq. 4).

B. **Storage Mapper**: The simple storage mapper component of this application selects multiple different cloud storage service providers in regular interval for different types of loads and records the response time.
C. **Statistical Data Storage and Collector**: The statistical data storage and collector module collects the response time from different sources and generates a report for the all the services providers. The parameters considered in the resultant dataset are amount of data in the load, network speed, and type of action on the service provider data and the time for response.
D. **Cloud Services and Containers:** In this research we have considered multiple cloud service storage containers for the experiment. The configuration is demonstrated [Table 11].

Table 11: Cloud Service Provider Instance Configuration Details

Instance Type	Number of Units	Type of Architecture	Disk Space (GB)	RAM (GB)
Small	2	32 Bit	160	1.7
Medium	2	32 Bit	350	1.7
Large	4	64 Bit	850	7.5
Extra Large	5	64 Bit	1690	15
Extra Large - High Speed CPU	5	64 Bit	1690	15

V. Understanding Reed-Solomon Erasure

The most effective and popular framework under Erasure Coding is Reed-Solomon framework. The framework can be applied in case of

$n \leq 2^z$, where n denotes number of disks and z denotes number of customer data
….Eq 11

To understand the framework for 256 storage containers or disks are considered. For a 256 disks, a Reed – Solomon code can be defined and implemented using Galois Field Arithmetical GF (2^8). The coefficient "a" can be defined in various ways. The basic implementation of Reed – Solomon is Cauchy construction. To understand Cauchy construction, we select any n unique numbers in the space of GF(2^Z). Hence the selected n number are distributed in two sets called X and Y, where X contains m elements and Y contains k elements. Hence:

$$a(_{i,j}) = \frac{1}{x_i \oplus y_j} \text{ with the help of GF } (2^z) \qquad \text{….Eq 12}$$

The most important factor that makes Reed-Solomon framework to implement is the simplicity. In this framework selecting k and m is random and does not depend on any factors and can be selected independently. The performance can be questioned as the time complexity for performing an XOR operating is less compared to GF. However the modern processors rely on vector instruction sets for performing array based multiplication operation. Hence the reduction in time for computation can be achieved. Moreover with the improvement of latency time for the I/O devices and cache memory is also been improving to match with the highly complex Erasure Codes. The implementation of Reed – Solomon is simple as many open source solutions are readily available for storage solutions.

VI. Response Time Comparison

The response times recorded from multiple transactions on various data sources are recorded. The data load is tested on mentioned cloud service providers across multiple parameters as amount of data in the load, network speed, and type of action [19][20]. We document the finds here:

Table 12: Cloud Service Provider Instance Configuration Details

Service Provider	Data Load (GB)	Network Speed (MBPS)	Action Type	Response Time (Mins)
Dropbox	1	10	Write / Read	1.4
Google Drive	1	10	Write / Read	1.4
Hightail	1	10	Write / Read	1.4
OneDrive	1	10	Write / Read	1.4
SugerSync	1	10	Write / Read	1.4
Dropbox	1	20	Write / Read	0.7
Google Drive	1	20	Write / Read	0.7
Hightail	1	20	Write / Read	0.7
OneDrive	1	20	Write / Read	0.7
SugerSync	1	20	Write / Read	0.7
Dropbox	5	10	Write / Read	7
Google Drive	5	10	Write / Read	7
Hightail	5	10	Write / Read	7
OneDrive	5	10	Write / Read	7
SugerSync	5	10	Write / Read	7
Dropbox	5	20	Write / Read	3.5
Google Drive	5	20	Write / Read	3.5
Hightail	5	20	Write / Read	3.5
OneDrive	5	20	Write / Read	3.5
SugerSync	5	20	Write / Read	3.5
Dropbox	10	1000	Write / Read	1.2
Google Drive	10	1000	Write / Read	1.2
Hightail	10	1000	Write / Read	1.2
OneDrive	10	1000	Write / Read	1.2
SugerSync	10	1000	Write / Read	1.2

We closely observe there is no deviation in the response speed.

VII. Conclusions and Future Scope

The performance of all cloud service providers are analysed on a textual dataset, which is large in volume and the effect of the number of queries on the same dataset is studied. It is proven that there is a large difference for highly efficient latency time depending on the database used. We have also noticed that the latency time for the queries are heavily dependent on network speed or the network bandwidth, through which the services are accessed. But in both the cases we found that the effect of contexture is significant. Hence we conclude that the performance of multiple cloud service providers will be generating nearly same performance. Hence we understand the Erasure code framework and multiple variations to the same. We also consider their applications on major cloud storage service providers. We also consider the reasons that lead to failure of storage. We realize that the Erasure codes are very effective for replication and recovery process during storage failure. However we also identify that reduction in storage cost cannot be minimized over the Erasure Codes to a maximum efficiency.

The further research needs to be carried out on multiple clustered storage containers arrangements where the virtualization factor to be considered. The redundancy control also to be considered to demonstrate the actual response time in the future works.

References

[1]. J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, "Oceanstore: An Architecture for Global-Scale Persistent Storage," Proc. Ninth Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS), pp. 190-201, 2000.

[2]. P. Druschel and A. Rowstron, "PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility," Proc. Eighth Workshop Hot Topics in Operating System (HotOS VIII), pp. 75-80, 2001.

[3]. A. Adya, W.J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J.R. Douceur, J. Howell, J.R. Lorch, M. Theimer, and R. Wattenhofer, "Farsite: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment," Proc. Fifth Symp. Operating System Design and Implementation (OSDI), pp. 1-14, 2002.

[4]. A. Haeberlen, A. Mislove, and P. Druschel, "Glacier: Highly Durable, Decentralized Storage Despite Massive Correlated Failures," Proc. Second Symp. Networked Systems Design and Implementation (NSDI), pp. 143-158, 2005.

[5]. Z. Wilcox-O'Hearn and B. Warner, "Tahoe: The Least-Authority Filesystem," Proc. Fourth ACM Int'l Workshop Storage Security and Survivability (StorageSS), pp. 21-26, 2008.

[6]. D.R. Brownbridge, L.F. Marshall, and B. Randell, "The Newcastle Connection or Unixes of the World Unite!," Software Practice and Experience, vol. 12, no. 12, pp. 1147-1162, 1982.

[7]. R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, and B. Lyon, "Design and Implementation of the Sun Network Filesystem," Proc. USENIX Assoc. Conf., 1985.
[8]. M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable Secure File Sharing on Untrusted Storage," Proc. Second USENIX Conf. File and Storage Technologies (FAST), pp. 29- 42, 2003.
[9]. S.C. Rhea, P.R. Eaton, D. Geels, H. Weatherspoon, B.Y. Zhao, and J. Kubiatowicz, "Pond: The Oceanstore Prototype," Proc. Second USENIX Conf. File and Storage Technologies (FAST), pp. 1-14, 2003.
[10]. R. Bhagwan, K. Tati, Y.-C. Cheng, S. Savage, and G.M. Voelker, "Total Recall: System Support for Automated Availability Management," Proc. First Symp. Networked Systems Design and Implementation (NSDI), pp. 337-350, 2004.
[11]. A.G. Dimakis, V. Prabhakaran, and K. Ramchandran, "UbiquitousAccess to Distributed Data in Large-Scale Sensor Networks through Decentralized Erasure Codes," Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN), pp. 111- 117, 2005.
[12]. A.G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized Erasure Codes for Distributed Networked Storage," IEEE Trans. Information Theory, vol. 52, no. 6 pp. 2809-2816, June 2006.
[13]. M. Mambo and E. Okamoto, "Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts," IEICE Trans. Fundamentals of Electronics, Comm. and Computer Sciences, vol. E80-A, no. 1, pp. 54- 63, 1997.
[14]. M. Blaze, G. Bleumer, and M. Strauss, "Divertible Protocols and Atomic Proxy Cryptography," Proc. Int'l Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT), pp. 127-144, 1998.
[15]. G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage," ACM Trans. Information and System Security, vol. 9, no. 1, pp. 1-30, 2006.
[16]. Q. Tang, "Type-Based Proxy Re-Encryption and Its Construction," Proc. Ninth Int'l Conf. Cryptology in India: Progress in Cryptology (INDOCRYPT), pp. 130-144, 2008.
[17]. G. Ateniese, K. Benson, and S. Hohenberger, "Key-Private Proxy Re-Encryption," Proc. Topics in Cryptology (CT-RSA), pp. 279-294, 2009.
[18]. J. Shao and Z. Cao, "CCA-Secure Proxy Re-Encryption without Pairings," Proc. 12th Int'l Conf. Practice and Theory in Public Key Cryptography (PKC), pp. 357-376, 2009.
[19]. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACM Conf. Computer and Comm. Security (CCS), pp. 598-609, 2007.
[20]. G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. Fourth Int'l Conf. Security and Privacy in Comm. Netowrks (SecureComm), pp. 1-10, 2008