2015 IEEE 29th International Parallel and Distributed Processing Symposium

Improving Storage Availability in Cloud-of-Clouds with Hybrid Redundant Data Distribution

Bo Mao[†], Suzhen Wu[*ΦB], Hong Jiang[‡]
[†]Software School of Xiamen University, China
[*]Computer Science Department of Xiamen University, China
[Φ]State Key Laboratory of High-end Server & Storage Technology, Jinan, China
[‡]Department of Computer Science and Engineering, University of Nebraska-Lincoln, USA
[B]Corresponding author: suzhen@xmu.edu.cn

***Abstract*—With the increasing utilization and popularity of the cloud infrastructure, more and more data are moved to the cloud storage systems. This makes the availability of cloud storage services critically important, particularly given the fact that outages of cloud storage services have indeed happened from time to time. Thus, solely depending on a single cloud storage provider for storage services can risk violating the service-level agreement (SLA) due to the weakening of service availability. This has led to the notion of Cloud-of-Clouds, where data redundancy is introduced to distribute data among multiple independent cloud storage providers, to address the problem. The key in the effectiveness of the Cloud-of-Clouds approaches lies in how the data redundancy is incorporated and distributed among the clouds. However, the existing Cloud-of-Clouds approaches utilize either replication or erasure codes to redundantly distribute data across multiple clouds, thus incurring either high space or high performance overheads. In this paper, we propose a hybrid redundant data distribution approach, called HyRD, to improve the cloud storage availability in Cloud-of-Clouds by exploiting the workload characteristics and the diversity of cloud providers. In HyRD, large files are distributed in multiple cost-efficient cloud storage providers with erasure-coded data redundancy while small files and file system metadata are replicated on multiple high-performance cloud storage providers. The experiments conducted on our lightweight prototype implementation of HyRD show that HyRD improves the cost efficiency by 33.4% and 20.4%, and reduces the access latency by 58.7% and 34.8% than the DuraCloud and RACS schemes, respectively.**

***Keywords*-Cloud-of-Clouds; Availability; Replication; Erasure Codes; Cost Efficiency**

I. Introduction

With the increasing popularity and cost-effectiveness of cloud storage, many companies and organizations have moved or planned to move data out of their own data centers into the cloud. Typical usage examples include storing backup data and online digital media, such as the recent announcement by the United States Library of Congress to move its digitized content to the cloud [4] and Netflix's dependence on the Amazon S3 storage [5] for the storage of its content. However, solely depending on a particular cloud storage provider has a number of potentially serious problems. First, it can cause the so-called vendor lock-in problem for the customers [1], [7], which results in prohibitively high cost for clients to switch from one provider to another as elaborated in Section II-A. Second, it can cause service disruptions, which in turn will lead to SLA violation, due to cloud outages, resulting in penalties, monetary or other forms, for the service providers. Examples include a series of high-profile cloud outages in the year of 2013 for cloud providers, such as Amazon, Microsoft and Google [28], from a 5-minute failure that costs half a million dollars to a week-long disruption that costs an immeasurable amount of brand damage. From January to March 2014, DropBox has experienced two times of service outages [28]. Third, solely depending on a particular cloud storage provider can also result in possible increased service costs and data security issues, such as the data leakage problem [29]. Thus using multiple independent cloud providers, so called Cloud-of-Clouds, is an effective way to provide better availability for the cloud storage systems.

In a Cloud-of-Clouds, data redundancy is introduced to judiciously distribute data among the clouds. Thus, the redundant data distribution scheme is critically important for storage availability, performance, cost and space efficiency. Replication achieves the goals of availability and market mobility, but at a very high storage and bandwidth cost for large files. A more economical approach is to spread the data across multiple providers by introducing erasure-code redundancy to tolerate possible failures or outages, such as RACS [1] and NCCloud [16]. However, these schemes suffer from performance degradation due to the small updates over the networked storage because of the well-known write-amplification problem [12]. For example, a small update in the RACS system will incur a total of 4 accesses, including traffic of 2 reads and 2 writes over the network. Furthermore, a recent study conducted on Facebook's warehouse cluster [26], [27] reveals that more than 180TB of data is transferred through the top-of-rack switches everyday for RS-coded data recovery. In other words, the recovery operations consume a large amount of cross-rack bandwidth, thereby rendering the bandwidth unavailable for the foreground jobs. Thus, the data recovery process of the erasure-coded schemes will also incur significant network traffic due to the recovery I/Os in the cloud storage systems.

1530-2075/15 $31.00 © 2015 IEEE
DOI 10.1109/IPDPS.2015.47

When examining replication- and erasure-code-based schemes in the context of cloud storage systems, I/O performance and space efficiency are two important metrics. Given the explosive growth in data volume with big data analytics [31], [37], the I/O bottleneck has become an increasingly daunting challenge in terms of both performance and storage capacity. Besides the normal I/O performance, two additional performance-critical operations emerge in a Cloud-of-Clouds: the degraded read operation due to and the recovery operation from a single cloud service outage. Moreover, Recent IDC studies indicate that in the past five years the volume of data has increased by almost 9 times to 7ZB per year and a more than 44-fold growth to 35ZB is expected in the next ten years [32]. Managing the data deluge on storage to support (near) real-time data analytics becomes an increasingly critical challenge for Big Data analytics in the Cloud.

On the other hand, previous studies on the workload characteristics have shown that files are of mixed sizes with both small and large files [2], [30]. Moreover, file metadata accesses are much more frequent than file accesses and account for more than half of all the user operations [2], [30]. Thus, the performance of file metadata accesses is critical to the overall system performance and directly affects user experience. The recent studies, including RACS [1], DuraCloud [10], DepSky [7], NCCloud [16], and our own analysis detailed in Section II indicate that replication-based schemes are performance-friendly to small files and file metadata while erasure-code-based schemes are performance-friendly and cost-efficient to large files. This suggests that, a sensible data distribution scheme in the Cloud-of-Clouds should dynamically utilize replication and erasure codes based on different file characteristics. To address the important storage availability issue in the Cloud-of-Clouds, we propose a hybrid data distribution approach, called HyRD, by considering the workload characteristics. HyRD utilizes replication to store the small files and file system metadata, and erasure codes to store the large files on multiple cloud storage providers. By exploiting the workload characteristics and the diversity of cloud providers, both the advantages of erasure codes and replication are exploited and their disadvantages are alleviated. The extensive trace-driven experiments conducted on our lightweight prototype implementation of HyRD show that HyRD significantly outperforms RACS and DuraCloud in the I/O performance measure of average response times. Moreover, our evaluation and analysis results also show that HyRD achieves comparable or better cost and space efficiency.

The rest of this paper is organized as follows. Background and Motivation are presented in Section II. We describe the HyRD architecture and design in Section III. The performance evaluation is presented in Section IV. We review the related work in Section V and conclude this paper in Section VI.

II. Background and Motivation

In this section, we present some important observations drawn from previous and our analysis of the vendor lock-in problem of cloud storage, and the characteristics of the replication- and erasure-code-based redundant data distribution in Cloud-of-Clouds to motivate the HyRD study.

A. The vendor lock-in problem

The services provided by the cloud storage are diverse [1], [6]. The cloud storage providers offer different pricing with different performance characteristics, with some including extra features such as geographic data distribution, access through mountable file systems, or specific APIs. Changes in these features, or the emergence of new providers with more powerful and attractive characteristics, might compel some users to switch from one provider to another. However, moving from one provider to another one may be very expensive because the switching cost is proportional to the amount of data that has been stored in the original provider [1]. The more data has been stored in the original provider, the higher the switching cost will be paid to the bandwidth cost of data migration. This puts the users at a disadvantage, that is, when the cloud storage provider that has stored the user's data raises prices or negotiates a new contract less favorable to the user, the user has no choice but to accept because of the high switching cost, hence the so called *vendor lock-in problem* [1], [7].

Besides the possible increased prices or pressed unfavorable new contract, vendor lock-in can also lead to possible data loss or unavailability for the users if their cloud storage provider goes out of business or suffers a service outage. Despite of the strict Service-Level Agreements (SLAs) between the cloud provider and the user, service failures and outages do occur and are almost unavoidable. The cloud outages in 2013, while infrequent, showed that the service unavailability may last up to several hours and even days [28]. A study conducted by ESG (Enterprise Strategy Group) research has shown that about 58% of professionals in SMBs (Small and Medium Businesses) can tolerate no more than four hours of downtime before experiencing significant adverse effect [14]. More seriously, EMC's Disaster Recovery Survey in 2013 [11] has observed that the average cost per hour of downtime is much higher than ever before and 54% users suffered from lost data or service downtime, which further stresses the importance of the service/data availability in cloud storage systems.

To address the vendor lock-in problem induced by single individual cloud providers, a Cloud-of-Clouds solution is proposed in the literature [1], [7], [10], [16]. It redundantly distributes data across multiple providers by means of data redundancy schemes, such as replication and erasure codes. As a result, users can maintain their mobility while insuring against outages of a single individual cloud provider.

Table I
COMPARISON BETWEEN HYRD AND THE STATE-OF-THE-ART SCHEMES.

Schemes	Redundancy	Recovery	Performance	Cost
RACS [1]	Erasure Codes	Hard	Low for small updates	Low
DuraCloud [4], [10]	Replication	Easy	Low for large accesses	High
DepSky [7]	Replication	Easy	Low for large accesses	High
NCCloud [16]	Network Codes	Moderate	Low for small updates	Low
HyRD	Replication and erasure code	Easy	High	Low

B. Replication vs. erasure codes

Two common redundant data distribution methods used in Cloud-of-Clouds to achieve high availability of data are replication-based and erasure-code-based schemes. Although replication has the potential to increase availability and durability, it introduces two important challenges to system architects. First, system architects must increase the number of replicas to achieve high availability for large systems. For example, at least three replicas are required in Hadoop [15]. Second, the increased number of replicas introduces the extra bandwidth and storage overhead to the system, especially for large files. However, for the small files, replication is still an efficient way to provide the best performance with a small bandwidth and storage overhead. This is because small files only account for a tiny fraction of the bandwidth and storage capacity requirement, making replication on them profitable and productive considering the substantial performance benefits, both in the normal and recovery states.

An erasure code provides redundancy with much less space overhead than strict replication. Erasure codes divide an object into m fragments and recode them into a larger n fragments such that the original fragments can be recovered from a subset of the n fragments. The fraction $r = m/n$ is called the code rate. A rate r code increases the storage cost by a factor of $1/r$. For example, the RAID5 code can be described by an (m=4, n=5) erasure code. The key property of erasure codes is that the original object can be reconstructed from any m fragments. The main advantage of erasure codes is the high space efficiency with good availability. However, since any m correctly verified fragments must be used to reconstruct a given lost fragment, it will introduce two serious performance problems. One is the extra time required to record the redundancy information, especially for small files. Take RAID5 for example, a small-file update will induce two read operations and two write operations. The other is the large amount of network traffic required to reconstruct data when a cloud provider suffers an outage or fails. For the RAID5 example, the recovery of a lost small file on the downed/failed storage provider will require read traffic from all surviving cloud storage providers. On the other hand, erasure codes offer a particular advantage for large files in that their access latency is reduced by virtue of the parallel accesses among multiple cloud providers.

Table I summarizes the state-of-the-art redundant data distribution schemes. In general, replication provides better performance while erasure codes provide better storage efficiency. However, the former imposes extremely high bandwidth and storage overhead, while the latter does not provide the robustness and expected high access performance in the Cloud-of-Clouds particularly for large files. It therefore hints at the possibility of a certain combination of the two that tries to retain their respective advantages and while hiding their disadvantages to provide the most appropriate redundant data distribution scheme in Cloud-of-Clouds.

Knowing the workload characteristics is important for the storage system design. The previous studies have shown that more than 50% of files are smaller than 4KB [2] and metadata accesses are the most frequent kind [30], [33]. They also found that files whose size ranges from 3 MB to 9 MB accounts for more than 80% of the total storage capacity. These results reveal that large files account for a very large fraction (%80) of storage space occupation while representing a very small percentage (%10 to %20) of the total number of files in a storage system [2]. In contrast, small files that are 4 KB or smaller account for the most user accesses [2], [19]. Thus, small files and file metadata that is very small in size should be stored with a replication-based scheme and large files should be stored with an erasure-code-based scheme for the performance and cost efficiency considerations. These important observations, combined with the urgent need to address the availability problem of cloud storage systems, motivate us to propose HyRD. In HyRD, large files are distributed in multiple cost-efficient cloud storage providers with erasure-coded data redundancy while small files and file system metadata are replicated on multiple high-performance cloud storage providers. By exploiting the workload characteristics and the diversity of cloud providers, HyRD retains the desirable advantages of both the replication-based and erasure-code-based schemes while effectively addressing the vendor lock-in problem in the Cloud-of-Clouds.

III. THE DESIGN OF HYRD

In this section, we first outline the main design objectives of HyRD. Then we present its architecture overview, some design considerations and prototype implementation.

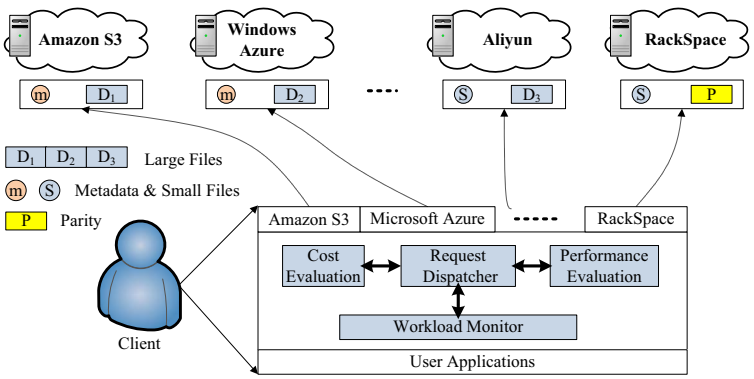


Figure 1. System architecture of HyRD.

A. The design of HyRD

The design of HyRD aims to achieve the following three objectives.

- *Improving the cloud storage availability* - By redundantly distributing user data in a Cloud-of-Clouds, it solves the vendor lock-in problem. With data redundancy schemes of replication and erasure codes, the service unavailability problem caused by the outage of single individual cloud storage providers is avoided.
- *Reducing the user access latency* - By using replication for the small files and file system metadata, the performance issue of update operations is avoided. Moreover, by using erasure codes for the large files, the access latency is reduced by exploiting the access parallelism across multiple cloud storage providers.
- *Improving the cost efficiency* - Since HyRD uses erasure codes to store the large files that occupy the most storage capacity, the overall storage efficiency is improved. Moreover, while small files and file system metadata account for most user accesses, they occupy disproportionally small capacity. Thus, replication on them does not increase overall capacity cost noticeably.

B. HyRD architecture overview

Figure 1 shows a system architecture overview of our proposed HyRD in the context of a Cloud-of-Clouds. Since more cloud storage services are provided by commercial cloud providers, the cloud providers are not allowed to execute users' code on the cloud storage side. As shown in Figure 1, HyRD resides on the client side and interacts with the cloud storage providers via their standard interfaces without any modification. Thus, HyRD can be easily applied to any cloud storage providers to use their cloud storage services.

HyRD has three main functional modules: Workload Monitor, Request Dispatcher, and Cost & Performance Evaluator. The *Workload Monitor* module is responsible for classifying the incoming write data into file metadata, large files and small files. The qualification of a file being large or small is workload independent but related to the access latency. We have conducted performance evaluations to select the best threshold to determine a file's type in Section IV. Based on the data type information (i.e., file system metadata, small file, or large file), the *Request Dispatcher* module decides which redundancy scheme should be used for the incoming data, and distributes the data to the corresponding cloud storage providers. The *Cost & Performance Evaluator* module is responsible for evaluating the cloud storage services from the perspectives of cost and performance. The cost characteristics of the cloud storage providers are summarized in Table II in Section IV and the performance characteristics are mainly described in terms of the access latency. These evaluation results will enable the Request Dispatcher module to select the appropriate cloud storage providers.

C. Design considerations

The data layout of the hybrid replication and erasure-codes distribution in a Cloud-of-Clouds introduces some design issues. We highlight the main ensuing design issues and our corresponding design choices.

Data distribution methods: The key idea of HyRD is to exploit the workload characteristics to choose either replication or erasure codes to distribute data among multiple cloud storage providers. At the present, HyRD only exploits the data type and file size characteristics. Besides file accesses, file system metadata blocks are critical to system performance. Before accessing a file, its metadata blocks must be loaded into the client memory. HyRD uses replication to store the file system metadata and groups the metadata in a directory together to exploit the access locality. For the file accesses, the file size is a critical parameter for HyRD to distribute data among multiple cloud storage providers. Since small files occupy a disproportionally small storage capacity and their updates are expensive in an erasure-code-based scheme, HyRD uses a replication-based scheme to store them. For large files that occupy a disproportionally large

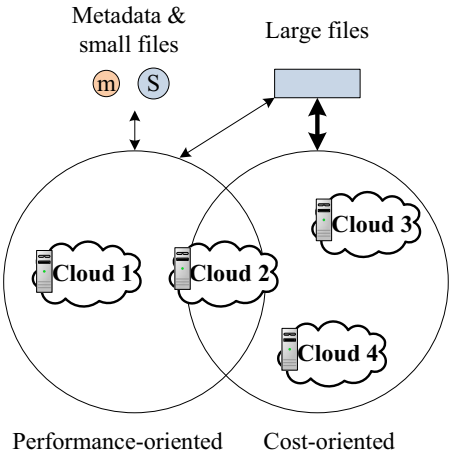


Figure 2. Data distribution between the performance-oriented and cost-oriented providers.

storage capacity and need parallel accesses to improve their performance, HyRD uses an erasure-code-based scheme to store them. However, how to distinguish a large file from a small file is nontrivial as it sensitively depends on a file-size threshold. We have conducted sensitivity experiments to investigate the file-size threshold, as shown in Section IV.

Evaluation of cloud storage services: The services offered by the cloud storage providers are diverse. Based on the design objectives of HyRD, it is important to evaluate the different cloud storage providers from the perspectives of performance and cost. In the HyRD evaluator module, the cloud storage providers are classified into two categories: performance-oriented providers where the data access latency is lower and cost-oriented providers where the storage capacity price is lower. For example, Aliyun is less expensive than Amazon S3 in storage price (per GB/month), as shown in Table II. A particular cloud storage provider can be in one category or both, as shown in Figure 2. Previous studies have shown that file system metadata and small files are accessed more frequently than large files [2]. Thus these data should be stored in performance-oriented providers for fast accesses. Because large files contribute to a disproportionally large storage capacity and thus the associated cost, HyRD puts them in the cost-oriented providers. To optimize performance of large files, some frequently accessed large files are also placed in performance-oriented providers, as shown in Figure 2.

Replication levels: The degree of data replication for file system metadata and small files within HyRD determines how resilient it is to cloud provider outages and failures, obviously the higher replication degree the more desirable. Unfortunately, higher degree of replication also comes with higher cost both in terms of storage space and access latency. Thus, there is a trade-off among resiliency, cost and performance. For example, higher degree of replication (i.e., more replicas) imply higher resiliency but also lower performance for write/update operations for file system metadata and small files. A recent survey of the cloud service outages indicates that two concurrent cloud outages are extremely rare [18]. This, combined with the known fact that high degree of replication significantly degrades system performance while also incurring high space cost, makes it sensible to choose the replication level of 2 in our current HyRD design. Nevertheless, it must be noted that the degree of replication in HyRD is configurable to satisfy the requirements of different users.

Recovery from service outage: An outage of cloud storage service is different from a disk failure in a disk array [36] in that the former results in a period of time during which cloud storage service is unavailable. The period may be hours and up to days. However, most outages will return to the normal state eventually. Thus, recovery in case of service outage in HyRD includes two phrases: (1) reconstruction on-demand during the unavailable period and (2) consistency update upon service's return to the normal state.

During the service unavailable period, all the write/update operations are performed as usual. For the update operations, the changes are logged; whereas, all the read operations are performed with on-demand read reconstruction. For the file system metadata and small files, the requested data is directly fetched from the replicated providers. The large files are reconstructed using the erasure-code redundancy. The unrequested data on the unavailable provider is not reconstructed and migrated to other storage providers. During the service unavailable period, the data on the off-line cloud storage provider may be invalid due to the write/update operations. Upon the unavailable provider's return to the normal state, the recorded write/update logs will perform the consistency updates on the returned provider to make the data consistent. When the logs are completely processed, the recovery process completes.

D. Prototype implementation

Our prototype is built as an independent module on the client side. To interact with multiple cloud storage providers, we have implemented a middleware of general cloud storage API, short for GCS-API. The GCS-API middleware hides the complexity of the cloud storage providers at the system level. Moreover, with such middleware, it is easy to add new cloud storage providers to the HyRD system.

Since each cloud storage service is modeled as a passive storage functional entity that supports five functions: List (lists the files of a container in the cloud), Get (reads a file), Create (creates a container), Put (writes or modifies a file in a container) and Remove (deletes a file). By passive storage functional entity, we mean that no operations other than what is needed to support the aforementioned five functions are executed. We assume that access control is

Table II
MONTHLY PRICE PLANS (IN US DOLLARS, $1 = ¥6.1) FOR AMAZON S3, WINDOWS AZURE STORAGE, ALIYUN OPEN STORAGE SERVICE AND RACKSPACE CLOUD FILES, AS OF SEPTEMBER, 10TH 2014 IN THE CHINA REGION.

Operations & Vendors	***Amazon S3 [5]***	***Windows Azure [35]***	***Aliyun [3]***	***RackSpace [25]***
Storage (per GB/month)	$0.033	$0.157	$0.029	$0.13
Data In (per GB)	Free	Free	Free	Free
Data Out to Internet (per GB)	$0.201	Free	$0.123	Free
Put, Copy, Post, and List (per 10K transactions)	$0.047	Free	$0.0016	Free
Get and others (per 10K transactions)	$0.0037	Free	$0.0016	Free
Category	Cost-oriented	Performance-oriented	Both	Cost-oriented

provided by the system in order to ensure that read requests are only allowed to invoke the List and Get functions. To easily use the various cloud storage services, HyRD uses the REpresentational State Transfer APIs (short for RESTful APIs) to perform the operations. RESTful APIs are application program interfaces (APIs) that use HTTP requests to perform the above five functions. RESTful APIs explicitly take advantage of HTTP methodologies defined by the RFC 2616 protocol. Besides the above five functions, the Evaluation module in HyRD will directly interact with the individual cloud storage providers to evaluate the corresponding values.

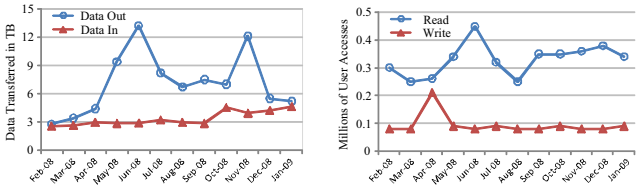
IV. PERFORMANCE EVALUATIONS

In this section, we first describe the experimental setup and evaluation methodology. Then we evaluate the performance of HyRD through extensive trace-driven experiments.

A. Experimental setup and evaluation methodology

Our tests are conducted in a desktop PC (client) with an Intel i5-3470 3.2 GHz quad-core processor, with 4GB of RAM and 1 Gigabit Ethernet connected to the China Education and Research Network [9]. Currently, our evaluations use the following four cloud storage providers in their default configurations: Amazon S3 [5], Windows Azure [35], Aliyun [3] and Rackspace [25]. Table II shows the monthly price plans for four major providers as of September 10th 2014. For all the cloud providers, we use the prices from the first chargeable usage tier in the China region (i.e., storage usage within 1TB/month in Amazon S3; the volume of data transferred out ranges between 1GB/month and 10TB/month).

Because cost analysis is a long-term evaluation, similar to RACS, we used a trace-driven simulation to understand the costs associated with hosting large digital libraries in the cloud. Our trace covers one year of activity on the Internet Archive (IA) servers [17] from Feb. 2008 to Jan. 2009. Figure 3 shows the amount of data written/read to/from the Internet Archive servers and the number of read/write requests issued to the Internet Archive servers during this one-year period. As shown in Figure 3, the volume of data transferred is dominated by reads that outweigh writes by ratio of 2.1:1 and read requests outnumber write requests by a ratio of 3.5:1. The trace represents HTTP and FTP



(a) Data transferred in TB (b) User read/write requests count

Figure 3. The amount of data written/read to/from and the number of read/write requests issued to the Internet Archive servers as a function of time during a one-year period.

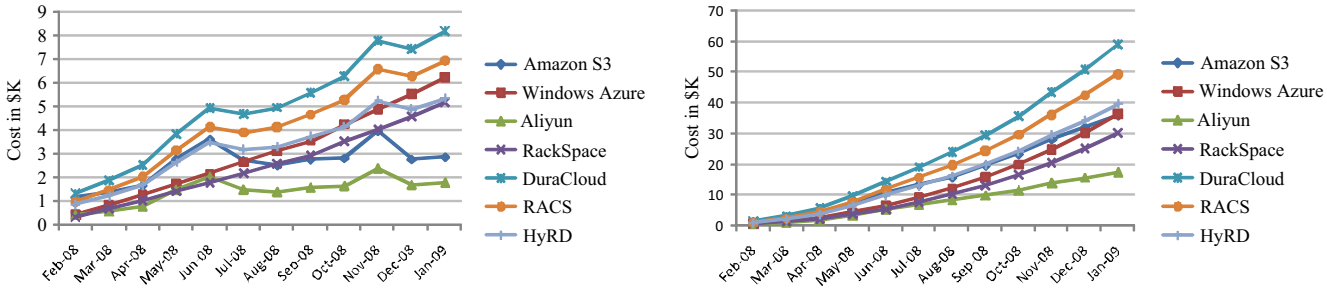
interactions that read and write various documents and media files (images, sounds, videos) stored at the Internet Archive and served to users. We believe that this trace is a good example of the type of workloads generated by online digital library systems, both in terms of the file sizes and request patterns.

To measure performance, we use the PostMark [13] benchmark tool to generate the file accesses as it is not practical to replay one year's trace. PostMark is designed to portray performance in desktop applications like electronic mail, netnews and web-based commerce, etc. We use PostMark to generate an initial pool of random text and image files ranging in size from a lower bound of 1024 bytes to a higher bound of 100M bytes. For erasure-coded redundancy, we choose the RAID5 scheme in HyRD as a case study to fairly compare with the RACS approach.

B. Cost simulation and analysis

In our cost simulation, it's assumed that the cloud services start with an empty storage without any data being preloaded. We estimate the cloud cost of moving the IA data to the cloud by using the up-to-date pricing schemes of the leading public cloud storage providers. Besides the bandwidth and storage costs, cloud providers also charge metadata operations, such as Put, Post, Post (short for 3Ps), List, Get and other operations based on per 10K transactions, as shown in Table II.

Figure 4(a) shows the estimated monthly cost of servicing the Internet Archive by using single-cloud storage providers (i.e., Amazon S3, Windows Azure, Aliyun, and Rackspace), the DuraCloud scheme that fully replicates all data between two cloud storage providers, and the RACS scheme and our HyRD scheme that both distribute data redundantly



(a) Monthly costs of different cloud storage providers

(b) Cumulative costs of different cloud storage providers

Figure 4. Estimated monthly and cumulative costs of hosting storage services on the cloud for different schemes.

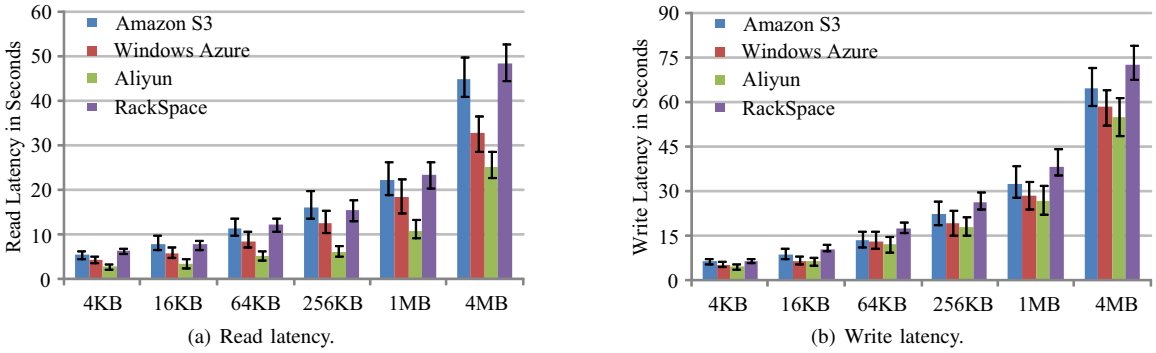
among four cloud storage providers. While RACS uses the RAID5 scheme to distribute all data, HyRD relies on a hybrid replication and RAID5 scheme to dynamically and adaptively distribute data based on their type and size. From Figure 4(a), we can see that the monthly costs of all the schemes, except for Amazon S3 and Aliyun, increase nearly monotonously. The reason is that with each additional month, the monthly cost not only includes storage cost and read cost of the current month, but also includes the storage cost of all previously written data. However, for the Amazon S3 and Aliyun providers, while their storage costs are lower than Windows Azure and RackSpace by more than 4 times, their own read costs are much higher than their storage costs. This means that for Amazon 3 and Aliyun their monthly bills are dominated by the read costs that can fluctuate from month to month. In other words, the monthly costs for the Amazon S3 and Aliyun providers depend much more on the read (data-out) operations than on write (data-in) operations.

Figure 4(b) shows the cumulative costs with different storage providers. First, we can see that DuraCloud is the most costly provider and Aliyun is the least costly provider. The high cost of DuraCloud comes from the full replication scheme that doubles the storage requirement and thus results in a storage cost that is the sum of those of the two involved individual providers. As expected, the cumulative storage cost increases from month to month as more data are accumulatively stored with time. Aliyun has the lowest cloud cost since it charges very little for the stored data and other operations, such as data out and metadata operations. Second, we see that the three Cloud-of-Clouds schemes (DuraCloud, RACS and HyRD) are more costly than the individual cloud storage providers. The reasons are twofold: (1) the Cloud-of-Clouds schemes add extra data redundancy that incurs additional storage cost; and (2) the update and write operations in the Cloud-of-Clouds will incur extra bandwidth cost due to the increased read operations. Third, the cloud cost of the HyRD scheme is 33.4% and 20.4% lower than that of the DuraCloud and RACS schemes, respectively. Compared with the DuraCloud scheme that uses full replication, both RACS and HyRD require less storage space overhead and thus achieve lower storage cost. Relative to the RACS scheme, our HyRD scheme requires less the storage cost by placing the many more large files in the cost-oriented cloud storage providers, such as Aliyun and RackSpac. Moreover, by reading data from the cost-oriented cloud storage providers, HyRD's cloud cost due to the data out operations is also reduced.

C. Performance results

In order to understand the performance of HyRD in a real deployment, we use the PostMark benchmark tool to run several workloads accessing a Cloud-of-Clouds composed of four popular single-cloud providers of Amazon S3, Windows Azure, Aliyun and RackSpace. Since the Internet bandwidth is not stable from time to time, we run each experiment for three times and use the average latency results with the deviation values. These experiments took place during about three months between July 5, 2014 and October 7, 2014.

We first evaluate the performance of individual single-cloud storage providers as a function of the request sizes of 4KB, 16KB, 64KB, 256KB, 1MB and 4MB, as shown in Figure 5. From the results presented in the figure, we can draw some interesting observations. First, Aliyun has the lowest access latency among the four single-cloud storage providers. This, combined with the fact that it has the lowest cloud cost as demonstrated in Figures 5 and 6, makes Aliyun a unique cloud storage provider in that it is both performance-oriented and cost-oriented and thus explains its categorization in last row of Table II. Second, there is a huge variance among the performance and the cost of the different cloud providers. It implies an important advantage of the Cloud-of-Clouds: we can exploit the workload characteristics and diversity of cloud storage providers to distribute data among multiple cloud storage providers. Third, when the file size increases from 1MB to 4MB, the access latency seems to increase disproportionally, which implies that the data transfer latency is disproportionally high at this level of file size and thus presents a clear gap the latency trend. Thus we set the file-size threshold at 1MB to distinguish large files from small files.



(a) Read latency. (b) Write latency.

Figure 5. Read/Write latency as a function of file size for single-cloud storage providers.

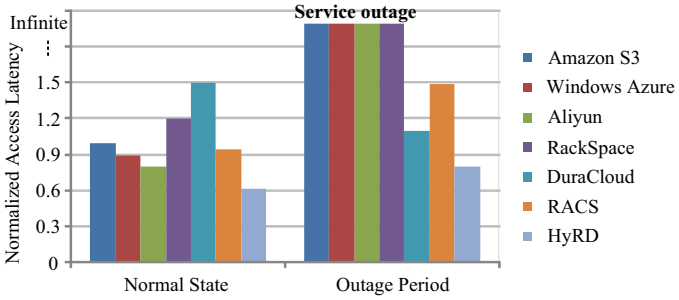


Figure 6. Access latency results of benchmark-driven experiments on real cloud deployment of the different schemes both in the normal state and in the service outage period of any one of the single-cloud storage providers. Note that the results are normalized to Amazon S3 and we set the Window Azure service off-line to emulate it's outage when evaluating the three Cloud-of-Clouds schemes.

Figure 6 shows the benchmark results in terms of access latency of the different schemes both in the normal state and in the service outage period of any one of the single-cloud storage providers. In the evaluations, we configure PostMark to issue files of size ranging from 1KB to 100MB to simulate a desktop PC client. We set the performance of a single-cloud Amazon S3 storage provider as the baseline. In the normal state, we see that HyRD performs the best among all the schemes. Its access latency is 58.7% and 34.8% lower than the DuraCloud and RACS schemes, respectively. Both RACS and HyRD distribute large files across multiple single-cloud storage providers, which enables them to exploit the access parallelism to improve the performance. However, for small files, the RACS scheme is less effective than HyRD. It is further validated by the performance results during the outage period when a single-cloud storage provider is off-line. For RACS, accessing (reading) the metadata or small files on the off-line provider will require it to access all the other three single-cloud storage providers to reconstruct the unavailable data. This will significantly increase the read traffic and decrease the bandwidth utilization. In contrast, the small-file/metadata access latency of neither DuraCloud or HyRD is noticeably affected by the service outage. The reason is that the metadata and small files are simply fetched from the surviving single-cloud storage provider that stores replicas of the unavailable data in the DuraCloud and HyRD schemes. In fact, the access latency of HyRD is 46.3% lower than that of RACS during the service outage period. Moreover, upon a service outage, the access latency of DuraCloud is better than that in the normal state since no double writes or updates are performed. This explains why the access latency of HyRD is only 27.3% lower than that of DuraCloud during the service outage period.

V. Related Work

As cloud storage becomes popular and cost efficient, more and more organizations and individual users will move their data to the cloud. Besides performance and security, availability of the cloud storage service is becoming increasingly more important for users. The notion of Cloud-of-Clouds is an effective approach to addressing the availability issue caused by the service outages of single-cloud storage providers.

There are several systems proposed for Cloud-of-Clouds. RACS [1] uses erasure coding to mitigate the vendor lock-in problem encountered by a user when switching cloud vendors. It transparently stripes data across multiple cloud storage providers with RAID-like techniques used by disks and file systems. HAIL [8] provides integrity and availability guarantees for stored data. It allows a set of servers to prove to a client that a stored file is intact and retrievable by the approaches adopted from the cryptographic and distributed-systems communities. NCCloud [16] achieves cost-effective repair for a permanent single-cloud provider failure to improve availability of cloud storage services. It is built on top of network-coding-based storage schemes called regenerating codes with an emphasis on storage repair, excluding the failed cloud in repair.

The above three systems are all based on erasure codes or network codes. In contrast, DuraCloud [10] utilizes replication to copy user content onto several different cloud storage providers to provide better availability. Moreover, it ensures

that all copies of user content remain synchronized. However, users will pay more money for the additional storage space and bandwidth required by DuraCloud. DEPSKY [7] improves the availability and confidentiality of commercial storage cloud services by building a Cloud-of-Clouds on top of a set of storage clouds, combining Byzantine quorum system protocols, cryptographic secret sharing, replication and the diversity provided by the use of several cloud providers. Different from these approaches, our proposed HyRD scheme takes the workload characteristics and the diversity of cloud storage providers, specially the file sizes, into the design of the redundant data distribution strategy so that the advantages of both the replication and erasure codes are exploited while hiding their disadvantages. As a result, both performance and storage efficiency are improved with the availability guarantee.

Integrating replication and erasure codes into one system is not a new idea. Our proposed HyRD takes inspirations from previous studies in the data organizations for RAID and file systems [20], [22], [23], [34]. For different RAID levels [24], replication-based disk array (RAID1) and parity-based disk arrays (RAID4/5) provide different performance and storage efficiency. HP AutoRAID [34] provides a two-level storage hierarchy inside a monolithic disk array controller. In the upper level of this hierarchy, RAID1 provides full redundancy and better performance. In the lower level, RAID5 parity protection is used to achieve lower storage cost. It automatically and transparently manages migration of data blocks between these two levels as access patterns change. Hot Mirroring [23] similarly combines RAID1 and RAID5 layouts, keeping hot data in the RAID1 portion and cold data in the RAID5 portion. It is a single-box solution and uses metadata to control the placement of data among disks comprising the disk array. In contrast to HP AutoRAID and Hot Mirroring, HyRD exploits the workload characteristics and the heterogeneity of cloud providers to choose between the replication redundancy and the erasure-coded redundancy to distribute data among multiple cloud storage providers, thus improving the availability of cloud storage services from the user's perspective.

VI. Conclusion

Availability of cloud storage services is one of the main factors that the users must consider seriously when deciding whether or not to move their data to the cloud. Depending on a single cloud storage provider has the inherent vendor lock-in problem that can potentially cost the user dearly. This paper proposed a hybrid redundant data distribution approach, called HyRD, by exploiting the workload characteristics and the diversity of cloud storage providers to improve the storage availability in Cloud-of-Clouds. In HyRD, large files are distributed in multiple cost-oriented cloud storage providers with the erasure-coded data redundancy while small files and file system metadata are replicated on multiple performance-oriented cloud storage providers. By exploiting the workload characteristics and the heterogeneity of cloud providers, both the advantages of erasure codes and replication are exploited while their disadvantages are alleviated. The experiments conducted on our lightweight prototype implementation of HyRD show that HyRD significantly outperforms existing Cloud-of-Clouds schemes, such as RACS and DuraCloud, in terms of the I/O performance and cost-effectiveness.

HyRD is an ongoing research project and we are currently exploring several directions for future research. First, we will apply data deduplication in the HyRD module to eliminate the redundant data and reduce the total data transferred over the network, thus further improving the performance and cost efficiency [21]. However, data deduplication requires powerful computing resources and extra memory space while HyRD is located in the client side. Applying data deduplication in HyRD is not easy and needs careful design considerations. Second, we will extend the HyRD design to consider the specific features of the diverse cloud storage services, thus further improving the flexibility of HyRD and the efficiency of cloud storage services.

VII. Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 61472336, No. 61402385 and No. 61100033, the US NSF Grant No. NSF-CNS-1116606 and NSF-CNS-1016609, National Key Technology R&D Program Foundation of China (No. 2015BAH16F02), the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, Fundamental Research Funds for the Central Universities (No. 20720140515). This work is also Supported by the State Key Laboratory of High-end Server & Storage Technology.

References

[1] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon. RACS: A Case for Cloud Storage Diversity. In *SOCC'10*, Jun. 2010.

[2] N. Agrawal, William J. Bolosky, John R. Douceur, and Jacob R. Lorch. A Five-Year Study of File-System Metadata. In *FAST'07*, Feb. 2007.

[3] Aliyun Open Storage Service. http://www.aliyun.com/.

[4] E. Allen and C. M. Morris. Library of Congress and DuraCloud Launch Pilot Program Using Cloud Technologies to Test Perpetual Access to Digital Content. In *Library of Congress, News Release. http://www.loc.gov/today/pr/2009/09-140.html*, Jul. 2009.

[5] Amazon S3. http://aws.amazon.com/s3/.

[6] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, and M. Zaharia. Above the Clouds: A Berkeley View of Cloud Computing. Technical Report No. USB/EECS-2009-28, EECS Department, University of California, Berkeley.

[7] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa. DepSky: Dependable and Secure Storage in a Cloud-of-Clouds. In *EuroSys'11*, Apr. 2011.

[8] K. D. Bowers, A. Juels, and A. Oprea. HAIL: A High-Availability and Integrity Layer for Cloud Storage. In *CCS'09*, Nov. 2009.

[9] China Education and Research Network. http://www.edu.cn/english_1369/index.shtml.

[10] DuraCloud Project. http://www.duracloud.org/.

[11] EMC Presents Disaster Recovery Survey 2013. http://emea.emc.com/microsites/2011/emc-brs-survey/index.htm.

[12] B. Fan, W. Tantisiriroj, L. Xiao, and G. Gibson. DiskReduce: RAID for Data-Intensive Scalable Computing. In *PDSW'09*, Nov. 2009.

[13] Filesystem Benchmarking with PostMark from NetApp. http://www.shub-internet.org/brad/FreeBSD/postmark.html.

[14] J. Gahm and J. Mcknight. Medium-size Business Server & Storage Priorities. *Enterprise Strategy Group*, Jun. 2008.

[15] Hadoop. http://hadoop.apache.org/.

[16] Y. Hu, H. Chen, P. Lee, and Y. Tang. NCCloud: Applying Network Coding for the Storage Repair in a Cloud-of-Clouds. In *FAST'12*, Feb. 2012.

[17] Internet Archive. http://www.archive.org/.

[18] O. Khan, R. Burns, James S. Plank, W. Pierce, and C. Huang. Rethinking Erasure Codes for Cloud File Systems: Minimizing I/O for Recovery and Degraded Reads. In *FAST'12*, Jan. 2012.

[19] J. Lofstead, M. Polte, G. Gibson, S. Klasky, K. Schwan, R. Oldfield, M. Wolf, and Q. Liu. Six Degrees of Scientific Data: Reading Patterns for Extreme Scale Science IO. In *HPDC'11*, Jun. 2011.

[20] Y. Ma, T. Nandagopal, K. Puttaswamy, and S. Banerjee. An Ensemble of Replication and Erasure Codes for Cloud File Systems. In *INFOCOM'13*, Apr. 2013.

[21] B. Mao, H. Jiang, S. Wu, and L. Tian. POD: Performance Oriented I/O Deduplication for Primary Storage Systems in the Cloud. In *IPDPS'14*, May 2014.

[22] B. Mao, H. Jiang, S. Wu, L. Tian, D. Feng, J. Chen, and L. Zeng. HPDA: A Hybrid Parity-based Disk Array for Enhanced Performance and Reliability. *ACM Transactions on Storage*, 8(1):1–20, 2012.

[23] K. Mogi and M. Kitsuregawa. Hot mirroring: a method of hiding parity update penalty and degradation during rebuilds for RAID5. In *SIGMOD'96*, Jun. 1996.

[24] D. Patterson, G. Gibson, and R. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). In *SIGMOD'88*, Jun. 1988.

[25] RackSpace. http://www.rackspace.com/cn/.

[26] K. Rashmi, N. Shah, D. Gu, H. Kuang, D. Borthakur, and K. Ramchandran. A Solution to the Network Challenges of Data Recovery in Erasure-coded Distributed Storage Systems: A Study on the Facebook Warehouse Cluster. In *HotStorage'13*, Jun. 2013.

[27] K. Rashmi, N. Shah, D. Gu, H. Kuang, D. Borthakur, and K. Ramchandran. A "Hitchhiker's" Guide to Fast and Efficient Data Reconstruction in Erasure-coded Data Centers. In *SIGCOMM'14*, Aug. 2014.

[28] The Worst Cloud Outages. http://www.cio.com/category/cloud-computing/.

[29] Tim Cook Says Apple to Add Security Alerts for iCloud Users, the Wall Street Journal, Sept. 2014. http://online.wsj.com/articles/tim-cook-says-apple-to-add-security-alerts-for-icloud-users-1409880977.

[30] A. Traeger, E. Zadok, N. Joukov, and C. Wright. A Nine Year Study of File System and Storage Benchmarking. *ACM Transactions on Storage*, 48(2):1–56, 2008.

[31] J. Tucci. Cloud + Big Data = Massive Change, Massive Opportunity, Keynote Address at UW CSE 2011-12 Annual Industrial Affiliates Meeting. Oct 2010.

[32] R. Villars, C. Olofson, and M. Eastwood. Big Data: What It Is and Why You Should Care, White Paper, IDC. 2011.

[33] Y. Wang, L. Alvisi, and Mike Dahlin. Gnothi: Separating Data and Metadata for Efficient and Available Storage Replication. In *USENIX ATC'12*, Jun. 2012.

[34] J. Wilkes, R. Golding, C. Staelin, and T. Sullivan. The HP AutoRAID hierarchical storage system. *Operating Systems Review*, 29(5):96–108, 1995.

[35] Windows Azure. http://www.windowsazure.cn/zh-cn/.

[36] S. Wu, H. Jiang, and B. Mao. IDO: Intelligent Data Outsourcing with Improved RAID Reconstruction Performance in Large-Scale Data Centers. In *LISA'12*, Dec. 2012.

[37] P. Zikopoulos, C. Eaton, D. Deroos, T. Deutsch, and G. Lapis. Understanding Big Data. 2010.