In Proceedings of 2016 Digital Media Industry & Academic Forum (DMIAF), Santorini, 2016, pp. 166-171. doi: 10.1109/DMIAF.2016.7574925

Securing Big Data in the Cloud by Protected Mapping over Multiple Providers

Chunming Rong*, Hongbing Cheng†, and Martin Gilje Jaatun‡
* Department of Electronic Engineering & Computer Science, University of Stavanger, Norway
†State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China
‡SINTEF ICT, Trondheim, Norway

***Abstract*—The Cloud is increasingly being used to store and process big data, and conventional security mechanisms using encryption are either not sufficiently efficient or not suited to the task of protecting big data in the Cloud. In this paper we present an alternative approach which divides big data among multiple Cloud providers, and instead of protecting the data itself, protects the mapping of the various data elements to each provider using a trapdoor function. Our initial analysis indicates that this is an efficient and secure approach for securing big data.**

***Index Terms*—Big Data Storage; Cloud Computing; Sharing; Virtual Mapping**

I. Introduction

Cloud computing [1] has emerged as a new computing and service paradigm, especially as a new design pattern for a particular type of big data centers. It offers the promise of simplified provisioning and management, lower costs, and access to resources that scale up and down with demand. At the same time, cloud computing has been proven to be a delivery platform to provide potential consumers with valuable information technology services over the Internet. In order to meet these services demands, cloud computing resources should be rapidly deployed and easily scaled.

In modern information technology, big data [2] is a term applied to "*data sets whose size is beyond the ability of commonly used software systems to store, manage, and process within a tolerable elapsed time*". Big data sizes are a constantly moving target as information technology develops, currently ranging from a few dozen terabytes to great petabytes of data in a data center. Generally, cloud computing mainly focuses on the storing and processing of big data, real-time data mining, and streaming media delivery etc. Data-intensive applications and research [3] will be integral to many future scientific endeavors, but will demand specialized privacy and security mechanisms [4] to make cloud computing platforms efficient and secure. In addition, the research community now has the option of accessing storage and computing resources on demand, and the IT industry is currently building multiple big data centers on cloud computing for social networks and applications. Consequently, large amounts of clients private and secret data (including meta-data) will be stored in cloud computing platforms, and will need protection during processing and transmission. Thus, cloud computing should be able to provide efficient security, access, and update mechanisms for the data [5]; not only for huge files running into petabytes, but also to small files that are only a few hundred bytes. In all the above cases, determining how to design a secure and efficient scheme for tenants to access their big data on cloud computing is crucial.

The rest of the paper is organized as follows: In Section II, related work for the security of big data is described. We propose a secure big data protection scheme for cloud tenants in Section III. In Section IV, we give a detailed analysis and comparison of the proposed scheme with other schemes. Conclusions are drawn in Section V.

II. Related Work

Some existing research has tried to address the topic of secure data storage in cloud computing. DepSky [6] was designed to improve the availability, integrity, and confidentiality of information stored in the cloud through the encryption, encoding, and replication of the data on diverse clouds that form a cloud-of-clouds. C. Wang et al. [7] proposed a flexible distributed storage integrity auditing mechanism, utilizing a homomorphic token and distributed erasure-coded data, analysis shows the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks. B. Wang et al. [8] utilize and uniquely combine the public key based homomorphic authenticator with random masking to achieve a privacy-preserving public cloud data auditing system, approaching cloud data storage security by resorting to an external audit party to check the integrity of outsourced data when needed. Singh et al. [9] propose using the RC5 encryption algorithm to protect the stored data in cloud computing, but just like any symmetric block cipher solution, this scheme cannot be applied efficiently when the data volume and variety are great.

In order to improve the cost and scalability benefits of cloud storage services, a federated Cloud storage system called MetaStorage [10] can integrate diverse Cloud storage providers. MetaStorage is a highly available and scalable distributed hash table that replicates data on top of diverse storage services. MetaStorage reuses mechanisms from Amazon's Dynamo for cross-provider replication and hence introduces a novel approach to manage consistency-latency tradeoffs by extending the traditional quorum (N,R,W) configurations to

an (N_P,R,W) scheme that includes different providers as an additional dimension.

Abu-Libdeh et al. [11] propose a RAID-like technique used by disks and file systems to store user data by striping across multiple providers in cloud computing, a proxy named RACS (Redundant Array of Cloud Storage) was designed to spread the storage load transparently over many providers. This technique aimed to reduce the cost of switching storage vendors for a large organization such as the Internet Archive by varying erasure-coding parameters.

Ousterhout et al. [12] argue for a new approach to datacenter storage called RAMCloud, where information is kept entirely in DRAM, and large-scale systems are created by aggregating the main memories of thousands of commodity servers. The authors believe that RAMClouds can provide durable and available storage and the combination of low latency and large scale will enable a new breed of data-intensive applications. Sammeta et al. [13] present a novel implementation of a Trust Enhanced Third Party Auditor (TETPA), which enables the Cloud Service Providers accountability, and protects the Cloud users benefits. The audit service can be used for dynamic integrity verification in multi cloud storage. This scheme is based on the techniques fragment structure, random sampling, index-hash table and Zero-Knowledge proofs, supporting provable updates to outsourced data and timely anomaly detection.

As the complexity, variety, and popularity of many advanced information services grows, cloud data centers have formed the backbone of these services offered via the Internet, including load-hosting, e-commerce, social networking, and a variety of more general services such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and other forms of grid/cloud computing. Some examples of these generic service platforms are Microsoft's Azure [14] platform, Google App engine [15], and Amazon EC2 [16]. Virtualization is the key to providing many of these services, and is being increasingly used within data centers to achieve better server utilization and more flexible resource allocation. However, virtualization also makes many aspects of data center management more challenging; the above cloud services provide high-bandwidth access to cost-effective storage and computing services. Cidon et al. proposed Copysets [17], a scalable replication technique to improve data durability while retaining the benefits of randomized load balancing.

Although the related work testifies to a steady progress in the field, there are still many key issues which should be studied deeply in order to make cloud big data more efficient and secure. From our point of view, this is especially true for secure storage and sharing mechanisms for cloud big data of tenants.

In addition to the conventional encryption schemes for protecting data centers mentioned above, various novel schemes have been proposed that employ variations on Rabin's n-out-of-m secret sharing scheme [18]. Proposals include those from Singh et al. [19], Parakh and Kak [20] and Luna et al. [21], but they all suffer from high computational overhead that will be prohibitive in a big data setting.

There are also a number of previous approaches to ensure security of data stored in distributed systems through dispersal of information. Jaatun et al. [22] propose a system for storing data at honest but curious cloud providers, but due to the extensive use of encryption on the storage path between cloud tenants and providers, their scheme is not suitable for big data (the same is true for the follow-up paper on this scheme [23]). Furthermore, their scheme does not provide a convenient way to share data with other users. Spillner et al. [24] present the NubiSave system, which offers a Redundant Array of Optimal Cloud Storage Providers (RAOC) for desktop users, this solution is also not suited to big data. Q. Wang et al. [25] study the problem of ensuring the integrity of data storage in Cloud Computing. In particular, the paper considers the task of allowing a third party auditor (TPA), on behalf of the cloud client, to verify the integrity of the dynamic data stored in the cloud. The introduction of TPA eliminates the involvement of the client through the auditing of whether his data stored in the cloud are indeed intact, which can be important in achieving economies of scale for Cloud Computing. Because the third party auditor (TPA) in cloud computing will be complicated and costly, this scheme must solve some important issues before practical applications.

III. The Proposed Mechanism

In cloud computing, big data storage services represent a basic function for their tenants. In the proposed mechanism, firstly, tenants' big data will be separated into many sequenced parts before storage, and then will be stored on different storage media owned by different cloud storage providers. When tenants access their data, the data parts in different data centers will be collected together and then be restored into original form based on the sequence number of each data part. Generally, the tenants' big data which is stored in cloud storage can be classified into public data and confidential data. There are no extra security requirements for public data, and each tenant can access these data freely; on the other hand, confidential data should always be kept secret and inaccessible to irrelevant persons or organizations.

As described above, the traditional network data security schemes cannot be efficiently applied to protect the cloud tenants' big data. Potential abuse of private information may for some tenants be a sufficiently serious threat to scare them away from cloud computing altogether. Currently, the storing of tenants' data on a cloud platform is a popular practice, and this is becoming more complicated and diversified than ever. In modern advanced information society, people have a variety of personalized requirements about their data information. Undoubtedly, privacy and security of personal data information is the most important concern for tenants when they store their confidential data on cloud storages.

A. Description of the basic proposed scheme

In order to make the confidential big data of tenants secure, we propose a secure cloud big data storage scheme based on a cryptographic virtual mapping of the big data. First,

big data will be pre-processed on the tenants side, and then distributed to cloud storage providers. Considering the huge cost of computing capability, the procedure of pre-processing can be a network middle-ware that runs on the cloud service side. During pre-processing, the big data will be separated into n sequenced parts first, and each part can be denoted by $part_i(i \in (1, n))$. These data parts will then be uploaded to cloud storage while indexing concurrently.

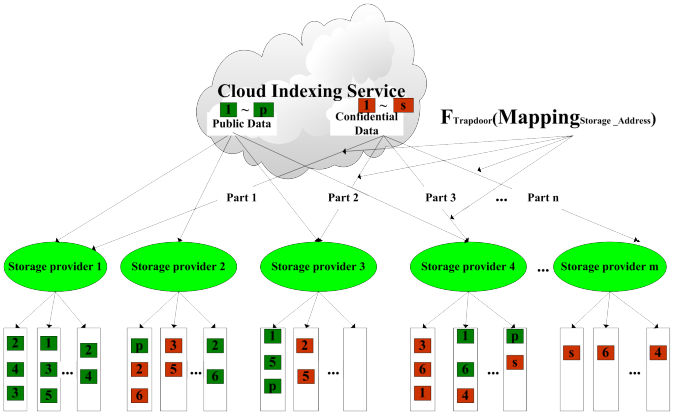

Fig. 1: Secure Cloud Storage for Tenants' Big Data

After all pre-processed sequenced data parts have been uploaded to cloud storage and indexed, they will be stored at m different storage providers, where each provider is identified as $provider_j(j \in (1, m))$. Evidently, n is always far greater than m, these m storage providers belong to different organizations, such as Google, Amazon and Yahoo. Each data part stored on a certain cloud storage provider will be allocated to some physical storage media that belongs to the storage provider. On the other side, in order to improve the robustness of the big data, we will store more than one copy of each data part on the cloud storage centers. In the practical application, the number of copies can be decided by the importance of the big data and the robustness of the storage system. In fact, we must balance the two aspects in our proposed scheme. Generally, we can divide the big data or big data set into data parts according certain principles, such as data types or Internet Protocol data packets. The detailed big data partitioning method and storage process of our proposed scheme are described individually below in Algorithm 1 and Algorithm 2.

Algorithm 1 The principle for big data partitioning

```
Decomposer(BigData){
   for(BigData) {
       if(consists of some kind of data type)
       {
              Type-identify(Big Data);
               Package(each data type);
      }// partition big data according to data type
     else
     {
        Block(Big Data);
        Package(each data type);
        // partition big data according to IP-resembling packet
}}}
```

In Algorithm 1, the proposed scheme will choose either-or approaches to partition the big data according to the feature of the big data and practical requirements. Both of the partitioning approaches will sequentialize each data part and mark some necessary tags for them, such as, uniqueness, shared-ness etc.

Algorithm 2 The process of storing Big Data on providers

```
Store-Data(BigData){
    Decomposer(BigData);
    for(i=0;i<n ;i++) {
       Identify (Data part n)
       File(Sequenced Value(1..n))
    }//Identify/operate on data parts to create and save
      // file of data parts sequenced value
     for(i=0;i<n;i++){
        if(only need to store one copy){
           Distribute and upload each data
           part to available storage service
           providers randomly
        }
        else {
           Distribute and upload more copies
           of data parts to the storage
           service providers
}}}
```

When the tenants' big data have been stored on the cloud storage centers , it will form a unique storage path for the big data given as:

$$Mapping_{Storage\_Path} = \{BigData.((Provider_1(M_1, M_2 \cdots M_r); (P_2(M_1, M_2 \cdots M_S); \cdots (Provider_2(M_1, M_2 \cdots M_t)\} \quad (1)$$

where $Provider_j$ denotes the j^{th} storage provider, and M denotes the physical storage media that belongs to a certain provider. The diagram of the storage procedure is described in Fig. 1.

We first introduce the trapdoor function before describing the proposed mechanism. A trapdoor function is a function that is easy to compute in one direction, yet believed to be difficult to compute in the opposite direction (finding its inverse) without special information, called the "trapdoor". Trapdoor functions are widely used in cryptography. In mathematical terms, if f is a trapdoor function there exists some secret information y, such that given $f(x)$ and y it is easy to compute x.

In order to protect the confidential big data of a tenant (such as A) from unauthorized access, we only need to encrypt the storage path of the big data but not the big data itself, and then we can get a cryptographic value which can be called cryptographic virtual mapping of big data. Generally, the path of the big data is k bytes in size and can be stored, processed and transferred quickly and easily. At the same time, in order to protect the confidential big data, the proposed mechanism will encrypt the path of the big data using a trapdoor function. The encrypted storage path result is denoted as $F_{Trapdoor}(Mapping_{Storage\_Path})$, and we use $Info_A$ to denote the secret information kept by tenant A, and given $F_{Trapdoor}(Mapping_{Storage\_Path})$, and $Info_A$ it is easy to

recover $F_{Trapdoor}(Mapping_{Storage\_Path})$. $Info_A$ is always kept secret by tenant A.

In the proposed scheme, other tenants must be authenticated by tenant A before getting the secret information $Info_A$ from A. It is easy to share the secret information $Info_A$ with other tenants or organizations based on an identity encryption algorithm; if someone else, such as tenant (organization) B, wants access to the secret information $Info_A$ B must provide his/her identity number ID_B to allow A to calculate the key k_B. The private key k_B is calculated by an identity based encryption algorithm. Afterwards, A can share the secret information $Info_A$ with B using the private key k_B (Note: when A authenticates B using identity based encryption, $Info_A$ is the private key of A k_A).

B. Improvements of the proposed scheme

1) Distribution of the big data pre-process procedure: Generally, the pre-process procedure for big data plays a role of client application when we distribute it on the local terminal, and clients will upload their big data to the cloud storages through the interface of the pre-process procedure, actually, the applications on the clients terminal always suffer the limitations of computation capacity and network bandwidth. In order to obtain better efficiency and availability, we can distribute the pre-process procedure on the cloud side as well, and it will play a role of service middleware for cloud clients. In such mode, the cloud clients only need to call the pre-process service rather than transmitting huge volumes of data.

2) Updating the storage when big data updated: In our proposed scheme, we will consider the issue of how to deal with the update of big data after their storage on cloud storage centers. In this section, we will first discuss different situations of big data, and then design a suitable data update approach for big data.

In the modern society, big data can address a multitude of problems if we can manage it properly. The insights gleaned from big data can help organizations deepen customer engagement, optimize operations, prevent threats and fraud, and capitalize on new sources of revenue. Currently, more than 2.5 billion gigabytes of high-velocity data are created every day, which is in a variety of forms, such as social media information, posts and transaction data, information gathered in sensors and medical devices, videos and audios records. Generally, the concept of big data has different meanings in different situations. We can call it big data when a kind of data is greater than gigabytes, but in fact, when the big data is a kind of information that is made of different kinds of files, such as word files, video files, database files etc., strictly, we should call such data a "big data set" rather than big data.

As we have noted above, it is necessary to re-store the big data if the big data itself has been updated after storing on the cloud storage centers. In a practical application, we will deal with the problem of big data update according to two different situations. One extreme situation is that the big data is a single file, thus, our proposed scheme will re-store the big data when it has been updated. The other situation is that the big data is a data set, and the big data consists of different kinds of files. In this situation, we will separate the big data into data parts according their file type and store them on cloud storage centers, thus, our proposed scheme only need to re-store the data parts which have been updated. In this scheme, it is necessary to delete the outdated data on the cloud service providers, and at the same time the cloud tenants need to delete the index information of the outdated big data. The re-storing process is described in Algorithm 3.

Algorithm 3 The re-storing process of updated big data

```
ReStore (BigData){
   for(BigData){
       if (big data only single file) {
             Notify (CSP) \\Notify cloud service providers
                       \\ to delete  outdated data on them.
             Delete(Index) \\ Delete the storage index
                       \\ information of outdated data
                      \\ from local terminal
             Store-Data(BigData)
      }
      else {
         for(all types of data parts){
              Distinguish which data parts
              have been updated
              Identify(updated data parts)
              Notify(CSP)
              Delete(Index)
              Packed(updated data parts)
              Distribute(updated data parts)
}}}}
```

Actually, when big data that only has one single file changed, the whole file will change as well, so, it is necessary to re-store the big data on the cloud storage centers. In general, as big data, one single file will not update frequently. On the other side, if big data or big data set consists of many different types of files, we only need to re-store the data parts that have been changed; this restoring operation is well managed and easy to run under the practical application occasions.

IV. Analysis and Comparison

A. Efficiency of the proposed scheme

It is well known that even when we transfer big data locally, the overhead of the transfer is often unendurable. Currently, as an efficient and economical data processing and storage service model, cloud computing can provide its tenants with the best data management and service. Recently, many large companies or enterprises focus on problems related to their data processing and maintenance, these companies or enterprises often spend a lot of time and overhead on their data management, but the effects are sometimes frustrating. This is especially true when these companies or enterprises do not have much business correlation with data management, in such a situation, these companies or enterprises would like to find a reliable and professional provider to deal with their data storage, management and maintenance rather than do it by themselves.

Evidently, big data transmission and its remote storage may cause many problems, such as, the time of processing is too long, the system will crash occasionally, and some uncertain system failures may occur. Transfer of big data directly to a

cloud storage data center may be vulnerable to system crash or failure, due to the huge data sizes involved. However, in our proposed scheme, the big data of tenants will be divided into smaller data blocks, and these smaller data blocks will be stored in cloud storage media one by one. Because these data blocks are much smaller than the primitive big data, they are very efficient for remote-distance data transmission and storage. Under the same network conditions, the transmission failure probability of the proposed scheme is lower than the situation when the big data is stored and transmitted directly.

Considering the diversity of cloud storage service providers, the big data can only be stored at a single storage service provider when tenants directly store their big data in the cloud. This risks incurring several problems, such as storage service error, system crash, tenants not being able to access the stored data, or even big data loss. In our proposed scheme, the divided data blocks are stored in different storage media, as long as we choose a certain amount of redundant data backup strategy, even if some of the storage service failed, the data of tenants will not be affected. Therefore, the proposed scheme can avoid the risk of "putting all your eggs in one basket".

In the proposed scheme, it is easy to protect the tenants confidential big data from unauthorized access when the owner of the big data only needs to encrypt the storage path, which we can refer to as cryptographic virtual mapping of big data. It is very efficient when the tenant only encrypts the path of the big data, because it is only on the order of k bytes, where k is the number of partitions. In the proposed scheme, it is easy to share the big data with other tenants by distributing the secret information to other tenants based on some secret sharing scheme, such as an identity-based encryption algorithm.

B. Security of the proposed scheme

Now we perform a security analysis of the proposed scheme. Let x be an adversary who wants to acquire the storage path of the big data illegally. According to the proposed scheme, the adversary can observe the whole big data only when he/she gets the storage paths of all data blocks. So, we can judge the security of storing part of big data on different cloud storages only by computing the probability that x knows the storage paths all of data blocks.

In the proposed scheme, big data will be separated into n parts and these n parts will be stored at m different storage providers. Since there is no need to consider some extreme situations, we assume that the adversary x can know the storage paths of the parts of big data with probability $P, (0 < P < 1)$. So, in this case, the probability that adversary x may be able to determine all the storage paths of the parts of big data is P^m, and when the probability P is low and the number of providers is large, the probability P^m will be very low.

Next, we consider the probability that when we store more than one copy of the data parts on cloud storage service providers. Similarly, big data of the tenants will be separated into n parts and stored at m different storage providers; we assume the multiple copies of the data parts are q. In this case, the best situation for the adversary x to determine the storage paths of the data parts is each data part has only one copy on any storage provider. The probability in such situation is $P\frac{n}{\lceil nq/m \rceil}$. On the other hand, the worst situation for the adversary x to determine the storage paths of the data parts is each data part almost has more than one copy on each storage provider, so, the adversary x must acquire all the storage paths of the data parts, and the probability in such a situation is P^m as well.

From the above analysis, we can conclude that the probability that the adversary x can achieve all the storage paths of the data parts is very low in the general situation. Now, we can assume that the adversary x will attempt to gain the path mapping file of the big data when it cannot achieve all of the storage paths of the data parts. Even if this situation is true, the proposed scheme shows excellent performance because the adversary can only get the path mapping after passing the authentication from the owner of the big data. In the proposed scheme, we use the trapdoor function to encrypt the storage paths, and such trapdoor function is easy to compute in one direction, and almost impossible to compute in the opposite direction without the secret information which is kept by the big data owner.

C. Comparison with other schemes

As more and more big data of individuals, companies and enterprises are placed in the cloud, some concerns are beginning to arise just how safe such kind of environment is. Despite a lot of promises from the cloud providers, some tenants are still reluctant to deploy their big data in the cloud. Security is one of the major issues which reduces the growth of cloud computing and complications with data privacy and data protection continue to plague the market. Inbarani et al. [26] present a survey of the different security risks that pose a threat to the cloud, focusing on the different security issues and concerns that have emanated due to the nature of the service delivery models of a cloud computing system.

In our paper, the idea of storing tenants data among cloud storage providers is similar to the approach of RACS [11], but our proposed scheme is more secure and feasible for cloud tenants. The advantages of our proposed scheme are:

1) No need for an extra proxy to deal with data distribution in cloud storage providers.
2) We store more than one copy of data parts to ensure the robustness of cloud data. Note: Data robustness means that our proposed scheme can improve the availability of the big data when some cloud storage providers fail, it can be calculated using available data parts
3) Security assurance for big data storage by encrypting the storage mapping.

We have compared some of the schemes in Section II with ours in some important aspects and the results are shown in Table I.

TABLE I: Comparison of the related storage scheme

Scheme	Suitable for big data?	Can data be shared?	Need security assumption?
Jaatun [22]	No	No	No
Spillner [24]	No	No	Yes
Wang [25]	Yes	Yes	Yes
Our scheme	Yes	Yes	No

V. Conclusion

Due to the enormous size of big data, owners need to consider the cost (both in terms of time and money) of encryption. Our presented solution avoids this by splitting the data among several cloud providers, and protecting the virtual mapping (needed to reconstruct/reassemble the big data) using a trapdoor function; at the same time, we consider the issue that how to deal with the update of big data after their being stored; and we proposed a suitable data update approach for big data storage. We analyze the efficiency and security of the proposed scheme and compare it with other related schemes and technology in different situations. The tentative results indicate that the proposed scheme is effective, secure and feasible for cloud tenants to store their big data.

Acknowledgements

This work was supported in part by the National Nature Science Foundation of China under Grant No. 61404213; the Six Kinds Peak Talents Plan project of Jiangsu Province under Grant No. 11-JY-009; and the Nature Science Foundation of Zhejiang Province under Grant No.LY14F020019, and in part by the European Commission through the EU FP7 project A4Cloud, grant nr. 317550.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599 – 616, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X08001957

[2] D. Kusnetzky. What is "Big Data?". [Online]. Available: http://blogs.zdnet.com/virtualization/?p=1708

[3] M. L. Norman and A. Snavely, "Accelerating data-intensive science with gordon and dash," in *Proceedings of the 2010 TeraGrid Conference*, ser. TG '10. New York, NY, USA: ACM, 2010, pp. 14:1–14:7. [Online]. Available: http://doi.acm.org/10.1145/1838574.1838588

[4] S. Pearson and G. Yee, Eds., *Privacy and Security for Cloud Computing*, ser. Computer Communications and Networks. Springer-Verlag London, 2013.

[5] X. Zhang, H. Du, J. Chen, Y. Lin, and L. Zeng, "Ensure data security in cloud storage," in *Network Computing and Information Security (NCIS), 2011 International Conference on*, vol. 1, may 2011, pp. 284 –287.

[6] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "Depsky: Dependable and secure storage in a cloud-of-clouds," *Trans. Storage*, vol. 9, no. 4, pp. 12:1–12:33, Nov. 2013. [Online]. Available: http://doi.acm.org/10.1145/2535929

[7] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Toward secure and dependable storage services in cloud computing," *IEEE Transactions on Services Computing*, vol. 5, no. 2, pp. 220–232, April 2012.

[8] B. Wang, B. Li, and H. Li, "Oruta: privacy-preserving public auditing for shared data in the cloud," *IEEE Transactions on Cloud Computing*, vol. 2, no. 1, pp. 43–56, Jan 2014.

[9] J. Singh, B. Kumar, and A. Khatri, "Improving stored data security in cloud using rc5 algorithm," in *2012 Nirma University International Conference on Engineering (NUiCONE)*, Dec 2012, pp. 1–5.

[10] D. Bermbach, M. Klems, S. Tai, and M. Menzel, "Metastorage: A federated cloud storage system to manage consistency-latency tradeoffs," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, July 2011, pp. 452–459.

[11] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon, "Racs: A case for cloud storage diversity," in *Proceedings of the 1st ACM Symposium on Cloud Computing*, ser. SoCC '10. New York, NY, USA: ACM, 2010, pp. 229–240. [Online]. Available: http://doi.acm.org/10.1145/1807128.1807165

[12] J. Ousterhout, P. Agrawal, D. Erickson, C. Kozyrakis, J. Leverich, D. Mazières, S. Mitra, A. Narayanan, G. Parulkar, M. Rosenblum, S. M. Rumble, E. Stratmann, and R. Stutsman, "The case for ramclouds: Scalable high-performance storage entirely in dram," *SIGOPS Oper. Syst. Rev.*, vol. 43, no. 4, pp. 92–105, Jan. 2010. [Online]. Available: http://doi.acm.org/10.1145/1713254.1713276

[13] N. Sammeta, R. Jagadeesh Kannan, and L. Parthiban, *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol II: Hosted by CSI Vishakapatnam Chapter*. Cham: Springer International Publishing, 2014, ch. Enhanced Trusted Third Party for Cyber Security in Multi Cloud Storage, pp. 525–533. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-03095-1_56

[14] Microsoft Azure. [Online]. Available: http://www.windowsazure.com/

[15] Google App Engine. [Online]. Available: https://developers.google.com/appengine/

[16] Amazon Elastic Compute Cloud. [Online]. Available: http://aws.amazon.com/ec2/

[17] A. Cidon, S. M. Rumble, R. Stutsman, S. Katti, J. Ousterhout, and M. Rosenblum, "Copysets: Reducing the frequency of data loss in cloud storage," in *Proceedings of the 2013 USENIX Annual Technical Conference*, 2013, pp. 37–48. [Online]. Available: http://dl.acm.org/citation.cfm?id=2535461.2535467

[18] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, pp. 612–613, November 1979. [Online]. Available: http://doi.acm.org/10.1145/359168.359176

[19] Y. Singh, F. Kandah, and W. Zhang, "A secured cost-effective multi-cloud storage in cloud computing," in *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, april 2011, pp. 619 –624.

[20] A. Parakh and S. Kak, "Online data storage using implicit security," *Information Sciences*, vol. 179, no. 19, pp. 3323 – 3331, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025509002308

[21] J. Luna, M. Flouris, M. Marazakis, and A. Bilas, "Providing security to the Desktop Data Grid," in *Parallel and Distributed Processing 2008 IPDPS 2008 IEEE International Symposium on*, 2008, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4536443

[22] M. G. Jaatun, G. Zhao, A. Vasilakos, Å. A. Nyre, S. Alapnes, and Y. Tang, "The design of a redundant array of independent net-storages for improved confidentiality in cloud computing," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 1, no. 1, p. 13, 2012. [Online]. Available: http://www.journalofcloudcomputing.com/content/1/1/13

[23] M. G. Jaatun, "I'll trust you – for now," in *Proceedings of International Conference on Internet of Things and Big Data*, 2016.

[24] J. Spillner, G. Bombach, S. Matthischke, J. Muller, R. Tzschichholz, and A. Schill, "Information dispersion over redundant arrays of optimal cloud storage for desktop users," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, dec. 2011, pp. 1–8.

[25] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, May 2011.

[26] W. S. Inbarani, G. S. Moorthy, and C. K. C. Paul, "An approach for storage security in cloud computing - a survey," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 2, no. 1, pp. 174–179, 2013.