DATA STORAGE SECURITY IN CLOUD COMPUTING FOR ENSURING EFFECTIVE AND FLEXIBLE DISTRIBUTED SYSTEM

Priyank Sirohi[1], Manav Bansal[2], Arpit Chhabra[3]
SCRIET, C. C. S UNIVERSITY, MEERUT

Abstract

This paper, proposes an effective and flexible distributed scheme with explicit dynamic data support to ensure the correctness of users data in the cloud. We rely on erasure correcting code in the file distribution preparation to provide redundancies and guarantee the data dependability. This construction drastically reduces the communication and storage overhead as compared to the traditional replication based file distribution techniques. By utilizing the homomorphism token with distributed verification of erasure coded data, our scheme achieves the storage correctness insurance as well as data error localization, whenever data corruption has been detected during the storage correctness verification, our scheme can almost guarantee the simultaneous localization of data errors, i.e. the identification of the misbehaving server. The new scheme further supports secure and efficient dynamic operations on data blocks, including: data update, delete and append. Extensive security and performance analysis shows that the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colliding attacks.

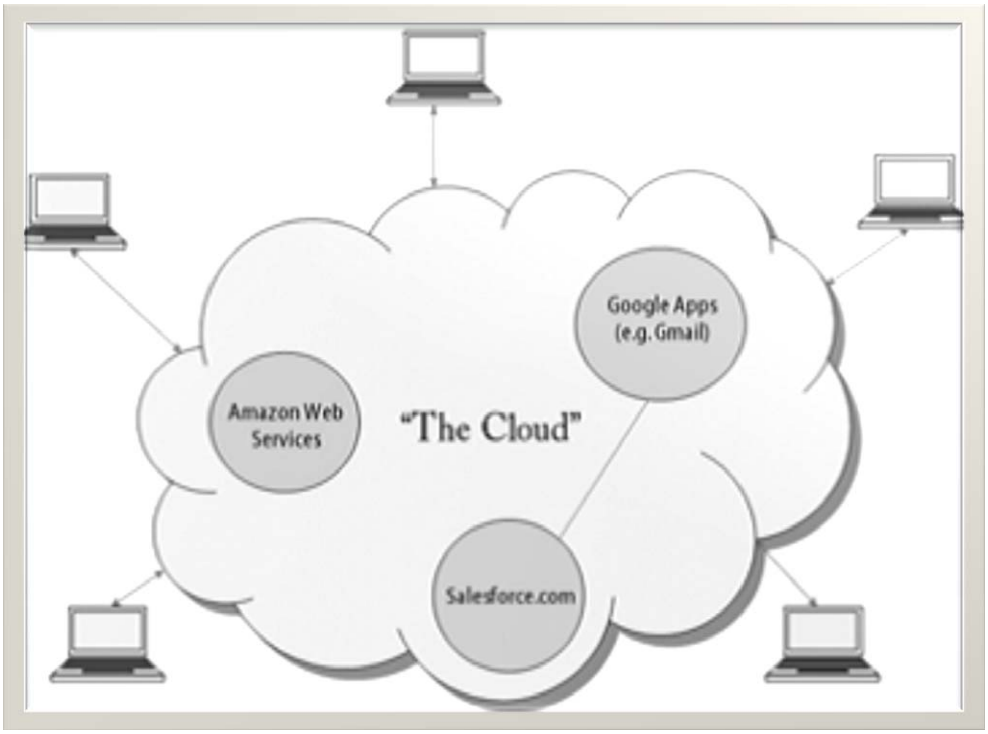
Key Words: Byzantine failure, homomorphic token, distributed scheme, cloud.

1. Introduction

Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. The ever cheaper and more powerful processors, together with the software as a service computing architecture are transforming data centers into pools of computing service on a huge scale. The increasing network bandwidth and reliable yet

Flexible network connections make it even possible that users can now subscribe high quality services from data and software that reside solely on remote data centers. Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management. The pioneer of Cloud Computing vendors, Amazon Simple Storage Service (S3) and Amazon Elastic Compute Cloud (EC2) are both well-known examples. While these internet-based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers for the availability and integrity of their data. From the perspective of data security, which has always been an important aspect of quality of service, Cloud Computing inevitably poses new challenging security threats for number of reasons. Firstly, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted due to the users' loss control of data under Cloud Computing. Therefore, verification of correct data storage in the cloud must be conducted without explicit knowledge of

the whole data. Considering various kinds of data for each user stored in the cloud and the demand of long term continuous assurance of their data safety.



.,

Fig.1 (a) Structure of a cloud

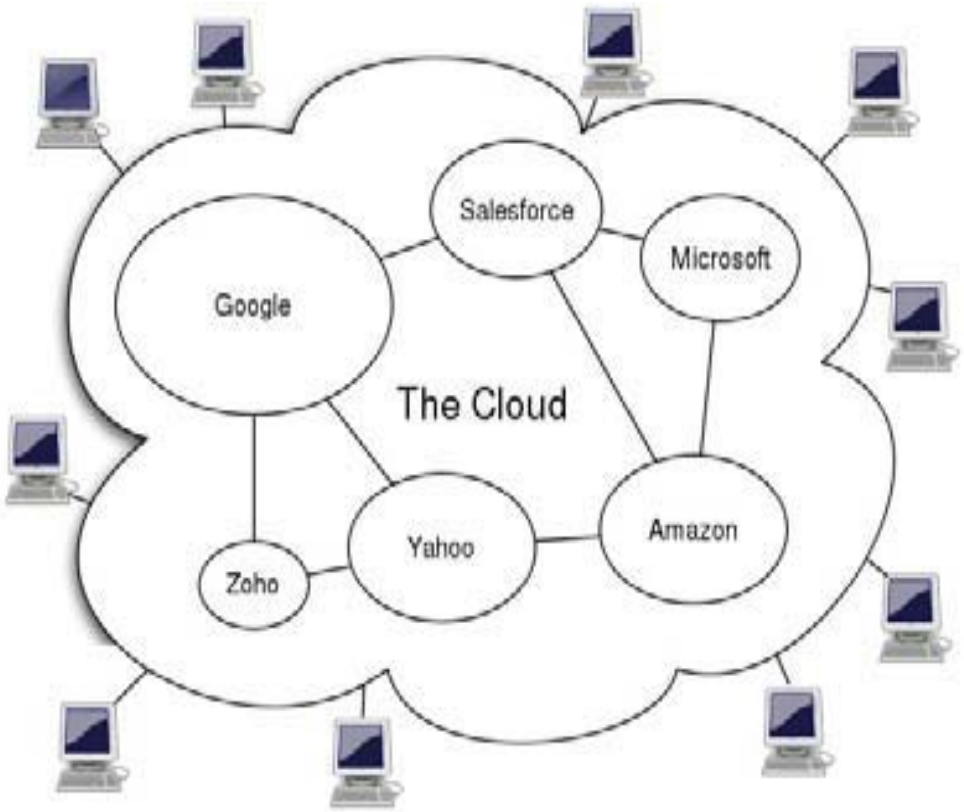


Fig.1(b) overview of cloud

The data stored in the cloud may be frequently updated by the users, including insertion, deletion, modification, appending, reordering, etc. To ensure storage correctness under dynamic data update is hence of paramount importance. However, this dynamic feature also makes traditional integrity insurance techniques futile and entails new solutions. Last but not the least, the deployment of Cloud Computing is powered by data centers running in a simultaneous, cooperated and distributed manner. Individual user's data is redundantly stored in multiple physical locations to further reduce the data integrity threats. Therefore, distributed protocols for storage correctness assurance will be of most importance in achieving a robust and secure cloud data storage system in the real world. However, such important area remains to be fully explored in the literature. Recently, the importance of ensuring the remote data integrity has been highlighted by the following research works. These techniques, while can be useful to ensure the storage correctness without having users possessing data, cannot address all the security threats in cloud data storage, since they are all focusing on single server scenario and most of them do not consider dynamic data operations. As a complementary approach, researchers have also proposed distributed protocols for ensuring storage correctness across multiple servers or peers. Again, none of these distributed schemes is aware of dynamic data operation and software that reside solely on remote data centers. Moving data into the cloud offers great convenience to
users since they don't have to care about the complexities of direct hardware management. The pioneer of Cloud Computing vendors, Amazon Simple Storage Service and Amazon Elastic Compute Cloud are both well-known examples. While these internet- based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers for the availability and integrity of their data From the perspective of data security, which has always been an important aspect of quality of service, Cloud Computing inevitably poses new challenging security threats for number of reasons. Firstly, traditional cryptographic primitives for the purpose of data security protection can not be directly adopted due to the users' loss control of data under Cloud Computing. Therefore, verification of correct data storage in the cloud must be conducted without explicit knowledge of the whole data. Considering various kinds of data for each user stored in the cloud and the demand of long term continuous assurance of their data safety, the problem of verifying correctness of data storage in the cloud becomes even more challenging. Secondly, Cloud Computing is not just a third party data warehouse as a result, their applicability in cloud data storage can be drastically limited.

2. Cloud Architecture

Cloud architecture, the systems architecture of the software systems involved in the delivery of cloud computing, typically involves multiple cloud components communicating with each other over loose coupling mechanism such as messaging queue. The two most significant components of cloud computing architecture are known as the front end and the back end. The front end is the part seen by the client,i.e., the computer user. This includes the client's network (or computer) and the applications used to access the cloud via a user interface such as a web browser. The back end of the cloud computing architecture is the cloud itself, comprising various computers, servers and data storage devices .Representative network architecture for cloud data storage is illustrated in Figure2.Three different network entities can be identified as follows: Users, who have data to be stored in the cloud and rely on the cloud for data computation, consist of both individual consumers and organizations. A CSP, who has significant resources and expertise in building and managing distributed cloud storage servers, owns and operates live

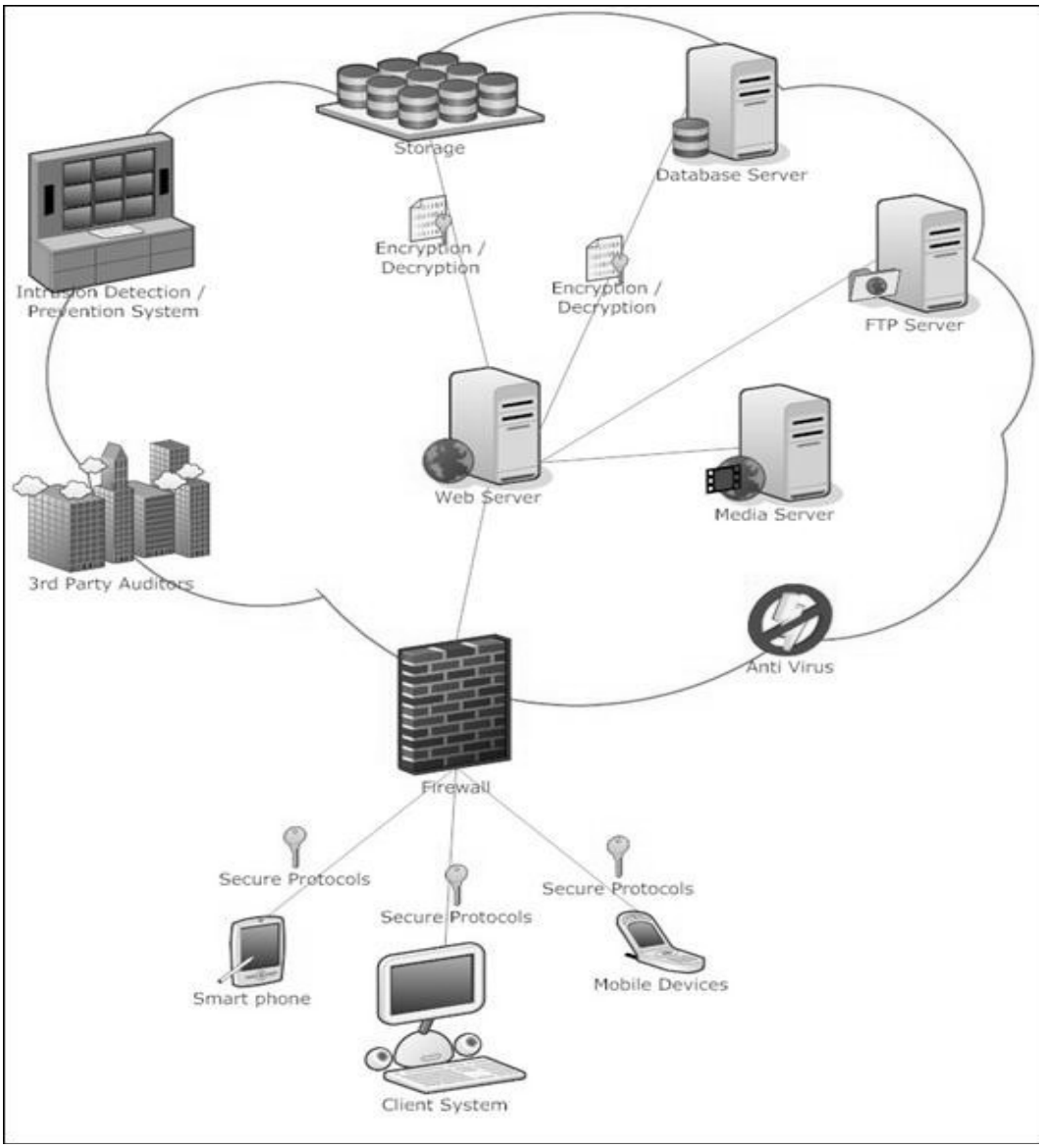


Fig. 2: Cloud data storage architecture

Cloud Computing system an optional TPA, who has expertise and capabilities that users may not have, is trusted to assess and expose risk of cloud storage services on behalf of the users upon request.

3. Design Methodologies

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity. Part of data design may occur in conjunction with the design of software architecture. More detailed data design occurs as each software component is designed. The architectural design defines the relationship between major structural elements of the software, the design patterns that can be used to achieve the requirements the system architecture, the interface representation, and the component level detail. During each design activity, we apply basic concepts and principles that lead to high quality

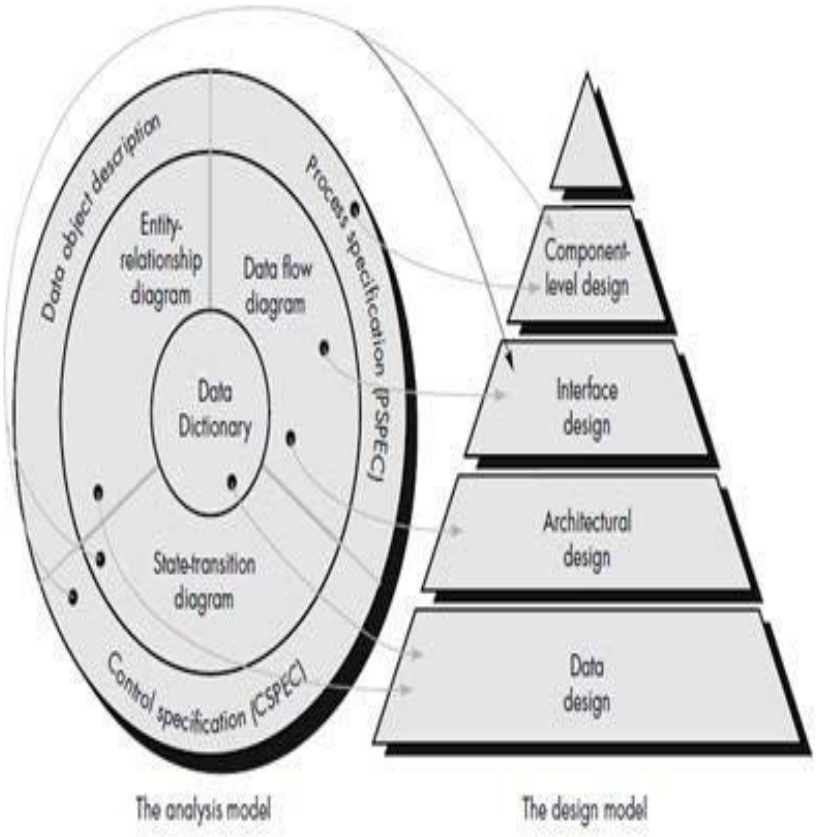


Figure.3 Translating Analysis model into Design model

The interface design describes how the software communicates within itself, with systems that interoperate with it, and with humans who use it. An interface implies a flow of information and a specific type of behavior. Therefore, data and control flow diagrams provide much of the information required for interface design. The component-level design transforms structural elements of the software architecture into a procedural description of software components. Information obtained from the PSPEC, CSPEC, and STD serve as the basis for component design. During design we make decisions that will ultimately affect the success of software construction and as important, the ease with which software can be maintained. To ensure the security and fidelity for cloud data storage under the aforesaid antagonist model, we aim to design efficient mechanisms for dynamic data verification and operation and achieve the following goals: (1) Storage correctness: to ensure users that their data are indeed stored appropriately and kept unharmed all the time in the cloud. (2) Fast localization of data error: to effectively locate the faulty server when data corruption has been detected. (3) Dynamic data support: to maintain the same level of storage correctness assurance even if users modify, delete or append their data files in the cloud. (4) Dependability: to enhance data availability against intricate failures, malevolent data modification and server colluding attacks, i.e. minimizing the effect brought by data errors or server failures. (5) Lightweight: to enable users to perform storage correctness checks with minimum overhead.

4. Ensuring Cloud Data Storage

In cloud data storage system, users store their data in the cloud and no longer possess the data locally. Thus, the correctness and availability of the data files being stored on the distributed cloud servers must be guaranteed. One of the key issues is to effectively detect any unauthorized data modification and corruption, possibly due to server compromise and/or random Byzantine failures. Besides, in the distributed case when such inconsistencies are

successfully detected, to find which server the data error lies in is also of great significance, since it can be the first step to fast recover the storage errors. To address these problems our main scheme for ensuring cloud data storage is presented in this section. The first part of the section is devoted to a review of basic tools from coding theory that is needed in our scheme for file distribution across cloud servers. Then, the homomorphism token is introduced. The token computation function we are considering belongs to a family of universal hash function chosen to reserve the homomorphism properties, which can be perfectly integrated with the verification of erasure-coded data. Subsequently, it is also shown how to derive a challenge response protocol for verifying the storage correctness as well as identifying misbehaving servers.

5 Security Analysis and Performance Evaluation

Our security analysis focuses on the adversary model as defined. We also evaluate the

6 Experimental Results

efficiency of our scheme via implementation of both file distribution preparation and verification token precipitation. In our scheme, servers are required to operate on specified rows in each correctness, verification for the calculation of requested token. We will show that this "sampling" strategy on selected rows instead of all can greatly reduce the computational overhead on the server, while maintaining the detection of the data corruption with high probability. Suppose nc servers are misbehaving due to the possible compromise or Byzantine failure. In the following analysis, we do not limit the value of nc, i.e., nc $\leq$ n. Assume the adversary modifies the data blocks in z rows out of the l rows in the encoded file matrix. Let r be the number of different rows for which the user asks for check in a challenge. Let X be a discrete random variable that is defined to be the number of rows chosen by the user that matches the rows modified by the adverse

Fig. 4 Server Login

Fig.5 Cloud Authentication Server

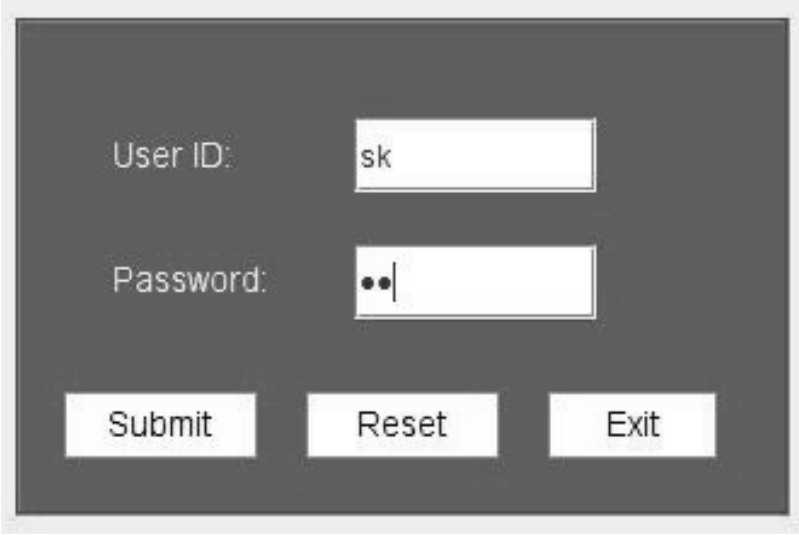


Fig.6 Client Login

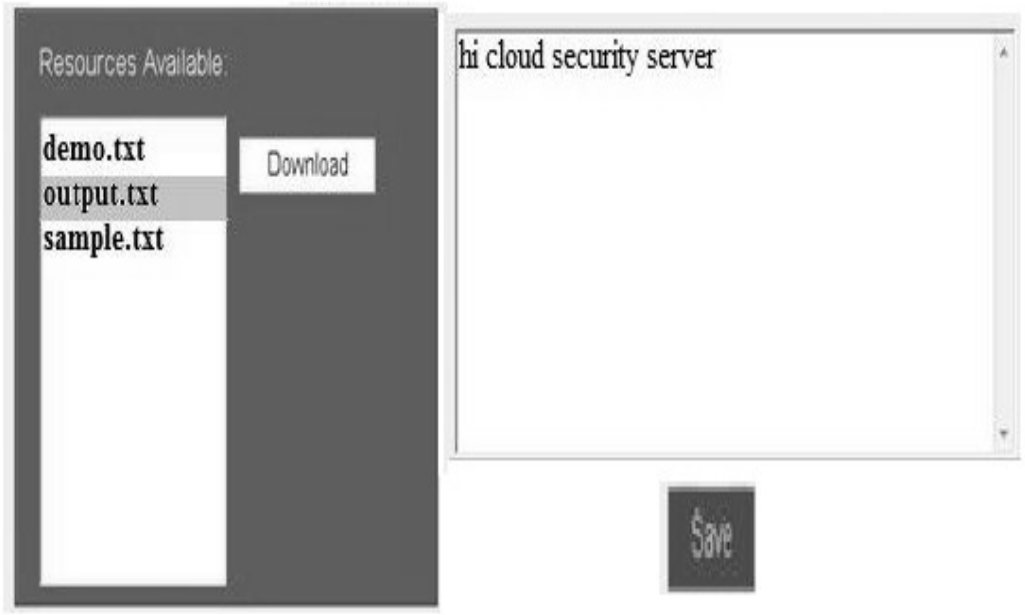


Fig.7 Availability of Resources

7. Conclusion

In this paper, we investigated the problem of data security in cloud data storage, which is essentially a distributed storage system. To ensure the correctness of users' data in cloud data storage, we proposed an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append. We rely on erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. By utilizing the homomorphism token with distributed verification of erasure coded data, our scheme achieves the integration of storage correctness insurance and data error localization, i.e whenever data corruption has been detected during the storage correctness verification across the distributed servers, we can almost guarantee the simultaneous identification of the misbehaving server(s). Through detailed security and performance analysis, we show that our scheme is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks. We believe that data storage security in Cloud Computing, an area full of challenges and of paramount importance, is still in its infancy now, and many research problems are yet to be identified. We envision several possible directions for future research on this area. The most promising one we believe is a model in which public verifiability is enforced. Public verifiability, supported in [6] [4], allows TPA to audit the cloud data storage without demanding users' time, feasibility or resources. An interesting question in this model is if we can construct a scheme to achieve both public verifiability and storage correctness assurance of dynamic data. Besides, along with our research on dynamic cloud data storage, we also plan to investigate the problem of fine-grained data error localization

8. References

1 Amazon.com "Amazon Web Services (AWS)" Online at http://aws. amazon.com.2008

2A. Juels and J. Burton S. Kaliski, "PORs: Proofs of Retrievability for Large Files," *Proc. of CCS '07*, pp. 584–597, 2007.

3H. Shacham and B. Waters, "Compact Proofs of Retrievability," *Proc.of Asiacrypt '08,* Dec. 2008.

4K. D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Cryptology ePrint Archive, Report 2008/175,2008, http://eprint.iacr.org/.

5G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z.Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," *Proc. Of CCS '07*, pp. 598–609, 2007.

6G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, Scalable and Efficient Provable Data Possession," *Proc. of SecureComm '08*, pp. 1– 10, 2008.

7 T. S. J. Schwarz and E. L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage," *Proc.of ICDCS '06*, pp. 12–12, 2006.

8M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, "A Cooperative Internet Backup Scheme," *Proc. of the 2003 USENIX Annual Technical Conference (General Track)*, pp. 29–41, 2003.

9 K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High- Availability and Integrity Layer for Cloud Storage," Cryptology ePrint Archive, Report 2008/489, 2008, http://eprint.iacr.org/.

10 L. Carter and M. Wegman, "Universal Hash Functions," *Journal of Computer and System Sciences*, vol. 18, no. 2, pp. 143–154, 1979.

11 J. Hendricks, G. Ganger, and M. Reiter, "Verifying Distributed Erasurecoded Data," *Proc. 26th ACM Symposium on Principles of Distributed Computing*, pp. 139–146, 2007.