Achieving Availability, Elasticity and Reliability of the Data Access in Cloud Computing

Lepakshi Goud.T[1]

Computer Science and Engineering, Department of P.G Studies,

Visvesvaraya Technological University, Belgaum, Karnataka, India

E-mail:lepakshigoudt@gmail.com[1]

Abstract--**Cloud computing is an elastic execution environment of resources involving multiple stakeholders such as users, providers, resellers, adopters and providing a metered service at multiple fine-granularities of the data access for a specified level of quality of service. To be more specific, a cloud is a platform or infrastructure that enables execution of code, services, applications etc., in a managed and elastic fashion. This paper is a main objective of achieve an availability, elasticity, reliability of the data access in cloud system, data access when users outsource sensitive data for sharing between owners and customers via cloud servers, which are not within the same trusted domain as data owners. Elasticity is an essential core feature of cloud systems and circumscribes the capability of the underlying infrastructure to adapt to potentially non-functional requirements, elasticity goes one step further, tough, and does also allow the dynamic integration and extraction of physical resources to the infrastructure. Whilst from the application perspective, this is identical to scaling, from the middleware management perspective this poses additional requirements, in particular regarding reliability is essential for all cloud systems in order to support today's data centre-type applications in a cloud, it is considered one of the main features to exploit cloud capabilities. This paper is proposes to denotes the capability to ensure constant operation of the system without disruption, i.e. No loss of data, no code reset during execution etc. We can achieve through redundant resource utilisation, many of the reliability aspects move from a hardware to a software-based solution ,there is a strong relationship between availability and reliability however, reliability focuses in particular on prevention of loss (of data or execution progress). Availability of services and data is an essential capability of cloud systems and was actually one of the core aspects to give rise to clouds in the first instance. Fault tolerance also requires the ability to introduce new redundancy in an online manner non-intrusively with increasing concurrent access, availability is particularly achieved through replication of data / services and distributing them across different resources to achieve load-balancing. This can be regarded as the original essence of scalability in cloud system.**

Keywords--Availability, Cloud computing, Data communication, Elasticity, Reliability, Computer Network.

I. INTRODUCATION

Today a growing number of companies have to process huge amounts of data in a cost-efficient manner. Classic representatives for these companies are operators of Internet search engines, like Google, Yahoo, or Microsoft. The vast amount of data they have to deal with every day has made traditional database solutions prohibitively expensive [1]. Instead, these companies have popularized an architectural paradigm based on a large number of commodity servers. Problems like processing crawled documents or regenerating a web index are split into several independent subtasks, distributed among the available nodes, and computed in parallel. In order to simplify the development of distributed applications on top of such architectures, many of these companies have also built customized data processing frameworks. Examples are Google's Map Reduce [2], Microsoft's Dryad [3], or Yahoo!'s Map-Reduce-Merge [4]. They can be classified by terms like high throughput computing (HTC) or many-task computing (MTC), depending on the amount of data and the number of tasks involved in the computation [5]. Although these systems differ in design, their programming models share similar objectives, namely hiding the hassle of parallel programming, fault tolerance, and execution optimizations from the developer. Developers can typically continue to write sequential programs.

The processing framework then takes care of distributing the program among the available nodes and executes each instance of the program on the appropriate fragment of data. For companies that only have to process large amounts of data occasionally running their own data canter is obviously not an option. Instead, Cloud computing has emerged as a promising approach to rent a large IT infrastructure on a short-term pay-per-usage basis. Operators of so-called Infrastructure-as-a-Service (IaaS) clouds, like Amazon EC2 [6], let their customers allocate, access, and control a set of virtual machines (VMs) which run inside their data centres and only charge them for the period of time the machines are allocated. The VMs are typically offered in different types, each type with its own characteristics (number of CPU cores, amount of main memory, etc.) and cost. Since the VM abstraction of IaaS clouds fits the architectural paradigm assumed by the data processing frameworks described above, projects like Hadoop [7], a popular open source implementation of Google's Map Reduce framework, already have begun to promote using their frameworks in the cloud.

Only recently, Amazon has integrated Hadoop as one of its core infrastructure services [9]. However, instead of embracing its dynamic resource allocation, current data processing frameworks rather expect the cloud to imitate the

static nature of the cluster environments they were originally designed for. E.g., at the moment the types and number of VMs allocated at the beginning of a compute job cannot be changed in the course of processing, although the tasks the job consists of might have completely different demands on the environment. As a result, rented resources may be inadequate for big parts of the processing job, which may lower the overall processing performance and increase the cost. In this paper we want to discuss the particular challenges and opportunities for efficient parallel data processing in clouds and present Nephele, a new processing framework explicitly designed for cloud environments. Most notably, Nephele is the first data processing framework to include the possibility of dynamically allocating/ de allocating different compute resources from a cloud in its scheduling and during job execution. This paper is an extended version of [10]. It includes further details on scheduling strategies and extended experimental results.

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 (SaaS) computing architecture, are transforming data canters 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 canters. 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) [11] 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. Recent downtime of Amazon's S3 is such an example [12]. 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, 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. 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 [13]–[14]. These techniques, while can be useful to ensure the storage correctness without having users possessing data, can not 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 an complementary approach, researchers have also proposed distributed protocols [15]–[16] for ensuring storage correctness across multiple servers or peers. Again, none of these distributed schemes is aware of dynamic data operations. As a result, their applicability in cloud data storage can be drastically limited.

Cloud Computing has been envisioned as the next-generation architecture of IT enterprise, due to its long list of unprecedented advantages in the IT history: on-demand self-service, ubiquitous network access, location independent resource pooling, rapid resource elasticity, usage-based pricing and transference of risk [17]. As a disruptive technology with profound implications, Cloud Computing is transforming the very nature of how businesses use information technology. One fundamental aspect of this paradigm shifting is that data is being centralized or outsourced into the Cloud. From users' perspective, including both individuals and IT enterprises, storing data remotely into the cloud in a flexible on-demand manner brings appealing benefits: relief of the burden for storage management, universal data access with independent geo graphical locations, and avoidance of capital expenditure on hardware, software, and personnel maintenances, etc [18].
While these advantages of using clouds are unarguable, due to the opaqueness of the Cloud—as separate administrative entities, the internal operation details of cloud service providers (CSP) may not be known by cloud users—data outsourcing is also relinquishing user's ultimate control over the fate of their data.

As a result, the correctness of the data in the cloud is being put at risk due to the following reasons. First of all, although the infrastructures under the cloud are much more powerful and reliable than personal computing devices, they are still facing the broad range of both internal and external threats for data integrity. Examples of outages and security breaches of noteworthy cloud services appear from time to time.

II. RELATED WORK

Existing work close to ours can be found in the areas of ―shared cryptographic file systems" and ―access control of

outsourced data". In [19], Kallahalla et al proposed Plutus as a cryptographic file system to secure file storage on untrusted servers. Plutus groups a set of files with similar sharing attributes as a file-group and associates each file-group with a symmetric lockbox-key. Each file is encrypted using a unique file-blcok key which is further encrypted with the lockbox-key of the file group to which the file belongs. If the owner wants to share a file-group, he just delivers the corresponding lockbox-key to users. As the complexity of key management is proportional to the total number of file-groups, Plutus is not suitable for the case of fine-grained access control in which the number of possible ―file-groups" could be huge. In [20], Goh et al proposed SiRiUS which is layered over existing file systems such as NFS but provides end-to-end security. For the purpose of access control, SiRiUS attaches each file with a meta data file that contains the file's access control list (ACL), each entry of which is the encryption of the file's file encryption key (FEK) using the public key of an authorized user. The extension version of SiRiUS uses NNL broadcast encryption algorithm [21] to encrypt the FEK of each file instead of encrypting it with each individual user's public key. As the complexity of the user revocation solution in NNL is proportional to the number of revoked users, SiRiUS has the same complexity in terms of each meta data file's size and the encryption overhead, and thus is not scalable.

Ateniese et al [22] proposed a secure distributed storage scheme based on proxy re-encryption. Specifically, the data owner encrypts blocks of content with symmetric content keys. The content keys are all encrypted with a master public key, which can only be decrypted by the master private key kept by the data owner. The data owner uses his master private key and user's public key to generate proxy re-encryption keys, with which the semi-trusted server can then convert the cipher text into that for a specific granted user and full fill the task of access control enforcement. The main issue with this scheme is that collusion between a malicious server and any single malicious user would expose decryption keys of all the encrypted data and compromise data security of the system completely. In addition, user access privilege is not protected from the proxy server. User secret key accountability is neither supported. In [23], Vimercati et al proposed a solution for securing data storage on untrusted servers based on key derivation methods [24]. In this proposed scheme, each file is encrypted with a symmetric key and each user is assigned a secret key. To grant the access privilege for a user, the owner creates corresponding public tokens from which, together with his secret key, the user is able to derive decryption keys of desired files. The owner then transmits these public tokens to the semi-trusted server and delegates the task of token distribution to it. Just given these public tokens, the server is not able to derive the decryption key of any file. This solution introduces a minimal number of secret key per user and a minimal number of encryption key for each file. However, the complexity of operations of file creation and user grant/revocation is linear to the number of users, which makes the scheme un scalable. User access privilege accountability is also not supported.

III.BASIC CONCEPTS OF PAPER

A. Cloud computing

Since ―clouds" do not refer to a specific technology, but to a general provisioning paradigm with enhanced capabilities, it is mandatory to elaborate on these aspects. There is currently a strong tendency to regard clouds as ―just a new name for an old idea", which is mostly due to a confusion between the cloud concepts and the strongly related P/I/SaaS paradigms (see also II.A.2, but also due to the fact that similar aspects have already been addressed without the dedicated term ―cloud" associated with it (see also II). This section specifies the concrete capabilities associated with clouds that are considered *essential* (required in any cloud environment) and *relevant* (ideally supported, but may be restricted to specific use cases). We can thereby distinguish non-functional, economic and technological capabilities addressed, respectively to be addressed by cloud systems.

- **Non-functional aspects:** represent *qualities* or *properties* of a system, rather than specific technological requirements. Implicitly, they can be realized in multiple fashions and interpreted in different ways which typically leads to strong compatibility and interoperability issues between individual providers as they pursue their own approaches to realize their respective requirements, which strongly differ between providers [25]. Non-functional aspects are one of the key reasons why ―clouds" differ so strongly in their interpretation (see also II.B).
- **Economic considerations:** Is one of the key reasons to introduce cloud systems in a business environment in the first instance. The particular interest typically lies in the reduction of cost and effort through outsourcing and / or automation of essential resource management. As has been noted in the first section, relevant aspects thereby to consider relate to the cut-off between loss of control and reduction of effort. With respect to hosting private clouds, the gain through cost reduction has to be carefully balanced with the increased effort to build and run such a system.
- **Technological challenges:** implicitly arise from the non-functional and economical aspects, when trying to realize them. As opposed to these aspects, technological challenges typically imply a specific realization – even though there may be no standard approach as yet and deviations may hence arise. In addition to these implicit challenges, one can identify additional technological aspects to be addressed by cloud system, partially as a pre-condition to realize some of the high level features, but partially also as they directly relate to specific characteristics of cloud systems.

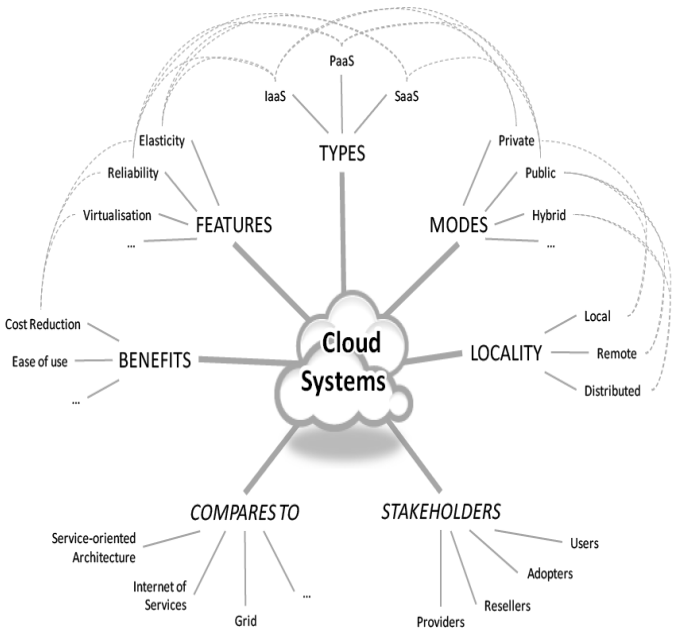

Fig.1: Structure of the Cloud computing

A.1 Availability

It services and data is an essential capability of cloud systems and was actually one of the core aspects to give rise to clouds in the first instance. It lies in the ability to introduce redundancy for services and data so failures can be masked transparently. Fault tolerance also requires the ability to introduce new redundancy (e.g. previously failed or fresh nodes) in an online manner non-intrusively (without a significant performance penalty). With increasing concurrent access, availability is particularly achieved through replication of data / services and distributing them across different resources to achieve load-balancing. This can be regarded as the original essence of scalability in cloud systems.

A.2 Elasticity

It is an essential core feature of cloud systems and circumscribes the capability of the underlying infrastructure to adapt to changing, potentially non-functional requirements, for example amount and size of data supported by an application, number of concurrent users etc [25]. One can distinguish between horizontal and vertical scalability, whereby *horizontal scalability* refers to the amount of instances to satisfy e.g. changing amount of requests, and *vertical scalability* refers to the size of the instances themselves and thus implicit to the amount of resources required maintaining the size. Cloud scalability involves both (rapid) up- *and* down-scaling. Elasticity goes one step further, tough, and does also allow the dynamic integration and extraction of physical resources to the infrastructure. Whilst from the application perspective, this is identical to scaling, from the middleware management perspective this poses additional requirements, in particular regarding reliability. In general, it is assumed that changes in the resource infrastructure are announced first to the middleware manager, but with large scale systems it is vital that such changes can be maintained automatically.

A.3 Reliability

It is essential for all cloud systems – in order to support today's data centre-type applications in a cloud, reliability is considered one of the main features to exploit cloud capabilities. Reliability denotes the capability to ensure constant operation of the system without disruption, i.e. no loss of data, no code reset during execution etc. Reliability is typically achieved through redundant resource utilisation. Interestingly, many of the reliability aspects move from hardware to a software-based solution.(Redundancy in the file systems vs. RAID controllers, stateless front end servers vs. UPS, etc.) Notably, there is a strong relationship between availability (see below) and reliability – however, reliability focuses in particular on prevention of loss (of data or execution progress).

B. Data communication

The fundamental purpose of a communications system is the exchange of data between two parties. Data communications are the exchange of data between two devices via some form of transmission medium such as a wire cable [22]. For data communications to occur, the communicating devices must be part of a communication system made up of a combination of hardware (physical equipment) and software (programs). The effectiveness of a data communications system depends on four fundamental characteristics: delivery, accuracy, timeliness, and jitter.

- ***Delivery***: The system must deliver data to the correct destination. Data must be received by the intended device or user and only by that device or user.
- ***Accuracy***: The system must deliver the data accurately. Data that have been altered in transmission and left uncorrected are unusable.
- ***Timeliness***: The system must deliver data in a timely manner. Data delivered late are useless. In the case of video and audio, timely delivery means delivering data as they are produced, in the same order that they are produced, and without significant delay. This kind of delivery is called *real-time* transmission.
- ***Jitter***: Jitter refers to the variation in the packet arrival time. It is the uneven delay in the delivery of audio or video packets. For example, let us assume that video packets are sent every 3D ms. If some of the packets arrive with 3D-ms delay and others with 4D-ms delay, an uneven quality in the video is the result.

C. Computer network

Network is a collection of computer and devices interconnected by communications channels that facilitate communications among users and allow users to share resources [26]. Networks may be classified according to a

wide variety of characteristics. A computer network allows sharing of resources and information among interconnected devices

Computer networks can be used for a variety of purposes:

- **Facilitating communications:** Using a network, people can communicate efficiently and easily via email, instant messaging, chat rooms, telephone, video telephone calls, and video conferencing.
- **Sharing hardware:** In a networked environment, each computer on a network may access and use hardware resources on the network, such as printing a document on a shared network printer.
- **Sharing files, data, and information:** In a network environment, authorized user may access data and information stored on other computers on the network. The capability of providing access to data and information on shared storage devices is an important feature of many networks.
- **Sharing software:** Users connected to a network may run application programs on remote computers.

IV.OUR PROPOSED SCHEME

As a result, their applicability in cloud data storage can be drastically limited. In this paper, we propose 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(s). Our work is among the first few ones in this field to consider distributed data storage in Cloud Computing. Our contribution can be summarized as the following three aspects:

- Compared to many of its predecessors, which only provide binary results about the storage state across the distributed servers, the challenge-response protocol in our work further provides the localization of data error.
- Unlike most prior works for ensuring remote data integrity, the new scheme supports secure and efficient dynamic operations.
- On data blocks, including: 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 colluding attacks.

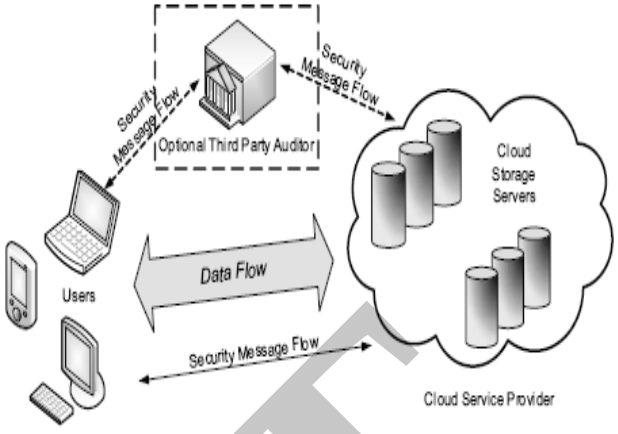

Fig.2: Cloud data storage architecture

these operations we are considering are block update, delete, insert and append. As users no longer possess their data locally, it is of critical importance to assure users that their data are being correctly stored and maintained. That is, users should be equipped with security means so that they can make continuous correctness assurance of their stored data even without the existence of local copies. In case those users do not necessarily have the time, feasibility or resources to monitor their data, they can delegate the tasks to an optional trusted TPA of their respective choices. In our model, we assume that the point-to-point communication channels between each cloud server and the user is authenticated and reliable, which can be achieved in practice with little overhead. Note that we don't address the issue of data privacy in this paper, as in Cloud Computing, data privacy is orthogonal to the problem we study here.

V. CONCLUSION

In this paper, we investigated the problem of data access and 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 reliability, elasticity in a 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 artily vectors and guarantee the data dependability and availability. By utilizing the homomorphism token with distributed verification of erasure recoded data, our scheme achieves the integration of to rage 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 and reliability 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 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.

REFERENCE

[1] R. Chaiken, B. Jenkins, P.-A. Larson, B. Ramsey, D. Shakib, S. Weaver, and J. Zhou. SCOPE: Easy and Efficient Parallel Processing of Massive Data Sets. Proc. VLDB Endow., 1(2):1265–1276 2008.

[2] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[3] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly. Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks. In EuroSys '07: Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007, pages 59–72, New York, NY, USA, 2007. ACM.

[4] H. chih Yang, A. Dasdan, R.-L. Hsiao, and D. S. Parker. Map-Reduce-Merge: Simplified Relational Data Processing on Large Clusters. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pages 1029–1040, New York, NY, USA, 2007. ACM.

[5] Raicu, I. Foster, and Y. Zhao. Many-Task Computing for Grids and Supercomputers. In Many-Task Computing on Grids and Supercomputers, 2008. MTAGS 2008. Workshop on, pages 1–11, Nov. 2008.

[6] Amazon Web Services LLC. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/, 2009.

[7] The Apache Software Foundation. Welcome to Hadoop! http: //hadoop.apache.org/ 2009.

[8] T. White. Hadoop: The Definitive Guide. O'Reilly Media, 2009.

[9] Amazon Web Services LLC. Amazon Elastic MapReduce.2009

[10] D. Warneke and O. Kao. Nephele: Efficient Parallel Data Processing in the Cloud. In MTAGS '09: Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers, pages 1–10, New York, NY, USA, 2009. ACM.

[11] Amazon.com, "Amazon Web Services (AWS)," Online at http://aws. amazon.com, 2008.

[12] N. Gohring, "Amazon's S3 down for several hours down for several hours.html, 2008.

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

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

[15] 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.

[16] 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/

[17] P. Mell and T. Grance, "Draft NIST working definition of cloud computing," Referenced on June. 3rd, 2009 Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index.html, 2009.

[18] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. UCB-EECS-2009-28, Feb 2009

[19] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Scalable secure file sharing on untrusted storage," in *Proc. of FAST'03*, 2003.

[20] E. Goh, H. Shacham, N. Modadugu, and D. Boneh, "Sirius: Securing remote untrusted storage," in *Proc. of NDSS'03*, 2003.

[21] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," in *Proc. of NDSS'05*, 2005.

[22] Gerard J. Holzmann "Data communications" AT&T Bell Laboratories, Murray Hill, New Jersey 07974, US.

[23] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Over- encryption: Management of access control evolution on outsourced data," in *Proc. of VLDB'07*, 2007

[24] M. Atallah, K. Frikken, and M. Blanton, "Dynamic and efficient key management for access hierarchies," in *Proc. of CCS'05*, 2005]

[25] The future of Cloud Computing Opportunities for EUROPEAN CLOUD COMPUTING BEYOND- 2010

[26] http://en.wikipedia.org/wiki/Computernetwork

