PUBLIC INTEGRITY AUDITING FOR SHARED DYNAMIC CLOUD DATA WITH MULTI USER REVOCATION

Mr. Mangesh B. Nagarkar, Prof. Ramesh G. Patole

[1] *Student, G.H.Raisoni College of Engineering & Management, Chas, Ahmednagar, Department of Computer Engineering, Maharashtra, India*
[2] *Professor, G.H.Raisoni College of Engineering & Technology, Wagholi, Pune, Department of Information Technology, Maharashtra, India*

ABSTRACT

The advent of the cloud computing makes storage outsourcing becomes a rising trend, which promotes the secure remote data auditing a hot topic that appeared in the research literature. Recently some research considers the problem of secure and efficient public data integrity auditing for shared dynamic data. However, these schemes are still not secure against the collusion of cloud storage server and revoked group users during user revocation In practical cloud storage system. In this paper, we figure out the collusion attack in the exiting scheme and provide an efficient public integrity auditing scheme with secure group user revocation based on vector commitment and verifier-local revocation group signature. We design a concrete scheme based on the our scheme definition. Our scheme supports the public checking and efficient user revocation and also some nice properties, such as confidently, efficiency, count ability and traceability of secure group user revocation. Finally, the security and experimental analysis show that, compared with its relevant schemes our scheme is also secure and efficient.

Keyword : - *Public integrity auditing, dynamic data, vector commitment, group signature, cloud computing*.

1. INTRODUCTION

The development of cloud computing motivates enterprises and organizations to outsource their data to third-party cloud service providers (CSPs), which will improve the storage limitation of resource con-strain local devices. Recently, some commercial cloud storage services, such as the simple storage service (S3) [1] on-line data backup services of Amazon and some practical cloud based software Google Drive [2], Dropbox [3], Mozy [4], Bitcasa [5], and Memopal [6], have been built for cloud application. Since the cloud servers may return an invalid result in some cases, such as server hardware/software failure, human maintenance and malicious attack [7], new forms of assurance of data integrity and accessibility are required to protect the security and privacy of cloud user s data.

To overcome the above critical security challenge of today's cloud storage services, simple replication and protocols like Rabin's data dispersion scheme [8] are far from practical application. The formers are not practical because a recent IDC report suggests that data-generation is outpacing storage availability [9]. The later protocols ensure the availability of data when a quorum of repositories, such as k-out-of -n of shared data, is given. However, they do not provide assurances about the availability of each repository, which will limit the assurance that the protocols can provide to relying parties.

For providing the integrity and availability of re-mote cloud store, some solutions [10], [11] and their variants [12], [13], [14], [15], [16], [17], [18] have been proposed. In these solutions, when a scheme supports data modification, we call it dynamic scheme, otherwise static one (or limited dynamic scheme, if a scheme could only efficiently support some specified operation, such as append). A scheme is publicly verifiable means that the data integrity check can be performed not only by data owners, but also by any third-party auditor. However, the

dynamic schemes above focus on the cases where there is a data owner and only the data owner could modify the data. Recently, the development of cloud computing boosted some applications [19], [20], [21], where the cloud service is used as a collaboration platform.

2. PROBLEM FORMULATION

2.1 Cloud Storage Model

In the cloud storage model, there are three entities, namely the cloud storage server, group users and a Third Part Auditor (TPA).

Group users consist of a data owner and a number of users who are authorized to access and modify the data by the data owner. The cloud storage server is semi-trusted, who provides data storage services for the group users. TPA could be any entity in the cloud, which will be able to conduct the data integrity of the shared data stored in the cloud server. In our system, the data owner could encrypt and upload its data to the remote cloud storage server. Also, he/she shares the privilege such as access and modify (compile and execute if necessary) to a number of group users. The TPA could efficiently verify the integrity of the data stored in the cloud storage server; even the data is frequently updated by the group users. The data owner is different from the other group users, he/she could securely revoke a group user when a group user is found malicious or the contract of the user is expired.

Fig 1: Security problem of server proxy group user revocation

2.2 Threat Model and Security Goals

Actually, in cloud environment, we assume that the cloud storage server is semi-trusted. Thus, it is reasonable that a revoked user will collude with the cloud server and share its secret group key to the cloud storage server. In this case, although the server proxy group user revocation way [24] brings much communication and computation cost saving, it will make the scheme insecure against a malicious cloud storage server who can get the secret key of revoked users during the user revocation phase. Thus, a malicious cloud server will be able to make data m, last modified by a user that needed to be revoked, into a malicious data m. In the user revocation process, the cloud could make the malicious data m become valid.

3. TECHNOLOGY

3.1 New Framework

We consider the database DB as a set of tuple (x, mx), where x is an index and mx is the corresponding value. Informally, a public integrity auditing scheme with updates allows a resource-constrained client to outsource the

storage of a very large database to a remote server. Later, the client can retrieve and update the database records stored in the server and publicly audit the integrity of the updated data.

According to previous researches, the proposed framework of our public integrity auditing for shared dynamic cloud data with secure group user revocation is given as follows:

Setup(1k, DB):

Let the database be DB = (i, mi) for 1 i q and the database is shared by a group of n users with only one data owner.

1) The data owner run the key generation algorithm of vector commitment to obtain the public parameters VC.KeyGen(1k, q).

2) Run the key generation of verifier-local revocation to obtain the user keys and revocations (gsk, grt) VLR.KeyGen(1k, n), where gsk = (gsk[1], gsk[2]...gsk[n]) and an n-element vector of user revocation tokens grt.

3) Run the computing algorithm to compute commitment and auxiliary information (C, aux) VC.ComPP(c1, ..., cq). Let the current database modifier be group user s(0 s n 1), and (gsk[s], gpk) be the secret/public key pair of the group user. Let Ct = V C.ComPP(ct1, ..., ctq) be the commitment on the latest database vector, where t is a counter with 0 as its initial value.

4) Run the signing algorithm over the commitment C. Specially, for the t-th time the group user s(0 s n1), whose secret key is gsk[s], compute and output a signature t VLR.Sign(gpk, gsk[s], C(t 1), Ct, t). Then, sends the signature t to the cloud storage server. If t is valid, then the server computes C(t) = tCt. Also, the cloud storage server adds the information of (t) = (C(t 1), Ct, t, t) to aux.

5) Finally, set public key parameter P K = (pp, gpk, C(t 1), C(t))

Query(P K, PP, aux, DB, i):

1) A group user run the opening algorithm to com-pute a proof i VC.OpenPP(ci, i, aux), where i is the proof of the i-th committed message and return = (ci, i, (t)).

Verify(P K, RL, i,):

1) Parse = (ci, i, (t)).

If the signature is valid after running the algorithm VLR.Verify(gpk, RL, (T)). Then, run the verification algorithm of vector commitment 0, 1 VC.VerPP(C(t), t , ci, i, i). The algorithm accepts when it output 1, which means that i is a valid proof that Ct was created by a sequence c1, ...cq , such that c = ci. Otherwise, return an error Ł.

Update(i,):

1) A group user first queries and verifies the database to make sure the current database is valid. More precisely, the group user obtain Query(P K, PP, aux, DB, i) and check that Verify(P K, i,) = mi.

2) Run the update algorithm over the new data and output the updated commitment and the update information (C,U) VC.Update(C, m, m, i). ProofUpdate(C, j, ci, i, U):

1) A third part auditor can first verify that, compared with the stored counter t, the latest counter equals t + 1. Then, run the proof of update algorithm of vector commitment to compute an update proof j

VC.ProofUpdatePP(C, j , mi, i, U)

for the mes-sage at position j, such that j is valid with respect to C which contains m as the new message at position j. Here, U = (m, m, i) is the update information.

2) Verify the commitment C, and its corresponding proof i is also valid over message mi.

3.2 A Concrete Scheme

In this section, we provide a concrete scheme from vector commitment [25] and verifier-local revocation group signature [27].

Setup(1k, DB):

Let k be a security parameter and DB = (i, mi) for 1 i q be the database. The database DB = (i, mi) is shared by a group of n users with only one data owner. The message space is M = Zp.

1) Let G, GT be two bilinear groups of prime order p equipped with a bilinear map e : G G GT , and g be a random generator of G. Randomly choose z1, ..., zq R Zp. For all i = 1, ..., q, set hi = gzi . For all i, j = 1, ..., q, i =6 j, set hi,j = gzizj . The data owner runs the key generation algorithm of vectorcommitmentVC.KeyGen(1k, q) to obtain the public parameters PP = (p, q, G, GT , H, g, (hi)i[q], hi,j i,j[q],i=6j) and the message space M= Zp. By using a collision-resistant hash function H : 0, 1 Zp, our scheme can be easily extended to support arbitrary messages in 0, 1.
2) Run the key generation of verifier-local revo-cation VLR.KeyGen(1k, n). Let G1, G2 be cyclic group of prime order p, and g1 be a generator of G1 and g2 be a generator of G2. Consider bilinear groups (G1, G2) with isomorphism , where g1 (g2). Select R Zp and set w = g2 . For each user, generate an SDH tuple i i by selecting i R Z such that i , (A , x) x p +x =6 0 and setting Ai g11/(+xi). Then, set the group public key gpk = (g1, g2, w). The private key is a tuple gsk[i] = (Ai, xi). The revocation token cor-responding to a user s secret key is grt[i] = Ai. Finally, the algorithm outputs (gpk, gsk, grk). is only known to the private-key issuer (the data owner).
3) Run the computing algorithm VC.ComPP(m1, ..., mq) to compute commitment C = hm11 , hm22 , ..., hmqq and auxiliary information aux = (m1, ..., mq).
4) Employ hash functions H0 and H as random oracles, with respective ranges G22 and Zp. For the t-th time data updating, run the signing algorithm VLR.Sign(gpk, gsk[i], C(t 1), Ct, over the commitment. Assume that the input message is C(t1), Ct, t 0, 1. Then, pick a random nonce r R Zp and obtain generators (u, v) H0(gpk, C(t 1), Ct, t , r) G22 and compute their images in G1 with u (u) and v (v) . Select an exponent R Zp and compute T1 u and T2 Aiv. Set xi Zp. Pick blinding values r, rx, and r R Zp. Compute helper values R1 ur , R2 e(T2, g2)rx e(v, w)r e(v, g2)r and R3 T1rx ur . Compute a challenge value c H(gpk, (C(t 1), Ct, t), r, T1, T2, R1, R2, R3) Zp using H. Compute s = r + c, sx = rx + cxi, and s = r + c Zp. Finally, output a signa-ture t (r, T1, T2, c, s, sx, s). Then, sends the signature t to the cloud storage server. If t is valid, then the server computes C(t) = t Ct. Also, the cloud storage server adds the informa-tion of (t) = (C(t 1), Ct, t, t) to aux.
5) Set public key parameter P K = (pp, gpk, C(t 1), C(t)).

Query(P K, pp, aux, DB, i):

1) We assume that the current public key is P K = (PP, gpk, C(t 1), C(t)). A user runs the open- ing algorithm VC.OpenPP (cit, i, aux) to compute t = q mjt q mjt zi a proof i j1,j=6i hi,j = (j=1,j=6i hj) of the i-th committed message and return = (mti, ti, (t)).

Verify(P K, i,):

1) On input a group public key gpk, a purported signature t, and the message C(t 1), Ct, t, the auditor first verify whether the signature is valid.

2) If = (mti, i, (t)), run the verification algorithm of vector commitment VC.VerPP(Cit, cti, i, ti) to verify that the equation mt ? e(Ct/hi i , hi) = e(it, g) holds. The algorithm accepts when it outputs 1, which means that ti is a valid proof that Ct was created to a sequence m1, ...mq, such that m = mi.

4. PROPOSED SYSTEM

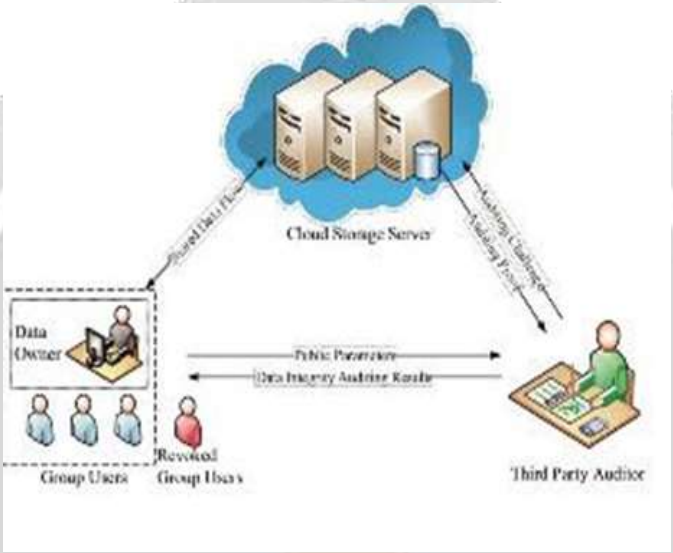

Fig 2: Proposed Architecture

System mainly focuses on patients history data retrieval at faster rate using NFC card .The system is introducing smart health record for ECG data based on reports classification is processed. Results are shown on graph. Likewise for other diseases also smartly record can be maintained. The Data mining Framework as follows.

Our scheme is designed to solve the security and efficiency problems of public data integrity auditing with multi-user modification, where the data has to be encrypted among a dynamic group and any group user can conduct secure and verifiable data update when necessary.

Some basic tools have been used to construct our scheme. Thus we assume that the underlying building blocks are secure, which include the vector commitment, group signature, and asymmetric group key agreement scheme. Based on this assumption, we show that our scheme is secure with respect to the following security analysis.

1) Security of Our Scheme: . Actually, Wu et al. [26] instantiated a one-round ASGKA scheme tightly reduced to the decision Bilinear Diffie-Hellman Exponentiation assumption in the standard model. Also, the security of adopted group signature scheme has been proved to be secure in the random oracle model. The security of the scheme is based on the strong Diffie-Hellman assumption and the Decision Linear assumption in bilinear groups as defined in Definition 1 and Definition 2. Thus, if we assume the two building blocks of our scheme is secure, then our scheme can be proven to be secure similar to [25].

If we assume there exists a polynomial-time ad-versary A that has a non-negligible advantage in the experiment for some initial database. Then we can use the adversary to build an efficient algorithm to break the Squ-CDH assumption in Definition 3. It means that the algorithm takes a tuple g, ga as input and output ga2 . Triv- ially, suppose that in a experiment where (i,) = (cit, it, (t)), the verify query output a value c =6Ł, c =6 ct ct , h)e(t, g) = the algorithm built by the adversary is /q.

2) Correctness of Our Scheme: If the server is assumed to be honest, then the proof = t (cit, it, (t)), where it = i=6j,1jq hi,jcj . Since ct ct cjt C(t)/Hthi i = Ct/hi i = i=6j,1jq hi,j , we have e(Ct/Hthciti , hi) = e(ti, g). Thus, the verification algorithm always output cti.

3) Efficiency of Our Scheme. : It is trivial that, except for the one time setup, the computational and storage overhead in our scheme invested by the group users are independent of the size of the data. More precisely, to verify the validity of the scheme, the verify algorithm run by the client requires only pairings and exponentiation in G. Also, in the update algorithm, the computation overhead of the client is indepen-dent of the size of data. The storage overhead of a group user is also independent of the size of data. We will pro-vide the detail efficiency and experiment analysis in the full version of this paper.

4) Countability of Our Scheme: Sine the update counter t is a public parameter, given the proof with the counter t, the client will firstly compare it with the public latest counter. if t = t, then the auditor verifies the corresponding signature t over t. Otherwise, if t 6= t the group auditor will reject the current result and report the malicious activity of the cloud storage server.

5) Traceability of Our Scheme. : The traceability of our scheme is based on the traceability of the adopted group signature. In the theorem 2 of reference [25], the authors provide the formal proof of the traceability of the group signature adopted. It means that if SDH is hard on (G1,G2), the group signature scheme is traceable.

5. PERFORMANCE EVALUATION

In this section, we provide both the numerical and the experimental analysis of our scheme and conduct the computation time cost comparison with [23] and [24].

5.1 Numerical Analysis

In this section, we conduct the numerical analysis of our scheme and compare the scheme with references [23] and [24]. First of all, all of the three schemes require one-time expensive computational effort in the Setup phase. Then, our scheme is secure against the collusion attack of the cloud storage server and the revoked users in the efficient scheme [24], and it is also efficient, since the computational resources invested by the client is independent on the size of the database. The reason is that, most of the expensive computation overhead is outsourced to the cloud storage server. Finally, the cloud storage server store all the database and its relevant materials. Thus, except some private key materials, the group users do not require to store any data locally.

We provide the time cost simulation for our scheme in different phases and the Table 1 presents the nu-merical analysis of computation of our scheme and two other schemes related. For the convenience of analysis, we denote by Mul a multiplication in G (G1, G2 and GT), Exp an exponentiation in G, Pair a computation of the pairing, and Hash a regular hashing operation. We omit other operations such as addition in G for all the schemes.

In the Query algorithm of our scheme, the computation overhead increases with the database item q. However, we need to remark that the server does not need to compute the proof each time. The reason is that the proof is identical for the same data item and the server only need to compute once for the first query on each index. Thus, the server could adopt some storage overhead to reduce the computational cost in the Query algorithm. Compare with scheme

[24], the Verify algorithm of our scheme bring much more computation overhead. The reason is that scheme [24] adopt the delegation technology for data updating. In our scheme, to prevent the attack against the collusion of the malicious and revoked group users, we adopt the group signature scheme with secure group user revocation. Although the Verify algorithm bring much more computational overhead than scheme [24], it is important that it is a constant part of our scheme. In the Update and ProofUpdate algorithms, the computation cost of our scheme and the scheme [24] grow with the increase of data elements (data items in our scheme). For the User Revocation algorithm, all the computation computation time cost grow with the increase of the challenge blocks (items) number.

The different is that, scheme [24] is efficient because the computation time cost will not grow with the increase of the group users. Thus, their scheme provide constant computation overhead with different group size. The computation time cost of our scheme grows with the revoked users number z, which is different from the scheme [23] growing with the increase of the group users number. Since it is reasonable that the revoked users number z is small than the group users d, we use $2z = d$ in our simulation.

5.2 Experimental evaluation:

In this section, we evaluate the thorough experimental evaluation of our scheme. Our experiments are simulated with the pairing-based cryptography library (PBC) on Linux Machine with Intelr CoreTM2 Duo Processor T9500 running at 2.60GHz and 3G memory. To precisely evaluate the computation complexity at different entities, we simulate all the entity on this machine.

The Query time cost of our scheme is linear with the data items number q, which will take approximately 4 seconds to query about 1000 data items. However, we need to emphasize that the computation cost is at the cloud storage server side, which is very powerful compare with the Linux system running on our laptop. Moreover, the server does not need to run the whole Query algorithm every time as analyzed in the previous section.

Actually, in the Verify algorithm, the computation overhead mostly comes from the group signature scheme. More precisely, to verify the validity of this phase, we need firstly to verify the integrity of the signature, which means that our scheme need to generate the time casted parameters such as R1, R2, and R3. Actually, the computation time cost of our scheme a constant number. Also, it is around 5 times that of the most efficient scheme [24]. In the Figure 5, we show the data update computation comparison with scheme [24], and both their computation overheads grow with the increase of the element number in each block.

In the PoofUpdate algorithm, the computation time cost grows with the increase of the elements number of each block and the number of selected challenging data blocks. Actually, the data blocks contain data elements in scheme [24] while not in our scheme. It is interesting that, if we do not consider the data elements in the scheme [24], the computation overhead of the two schemes in with the same challenging number are almost the same.

The User Revocation algorithm simulation in Fig-ure 7 shows that scheme [24] is the most efficient one. Compare with scheme [23] whose computation overhead grows rapidly with the increase of group users number and the selected challenging blocks number, scheme [24] and our scheme have a flat-ting growth. The reason is that, scheme [23] has to consider the whole group users number, while the computation time cost in our scheme is related to the revoked group users. It means that we need to verify ? $e(T2/A, u) = e(T1, v)$ for each user in the revocation list. The best scheme is [24], whose computation overhead is irrelevant to the group users number. They achieve this by allowing the cloud storage server to recomputed the authentication tag of blocks last modified by a revoked group user. We analyze this tag update delegation way in the previous section and point out that it is not secure against the cloud storage server and revoked group users collusion attack.

6. RESULTS

As we know, the comparison of computation cost is obvious. Our Update time is Exp , it is much lower than the update time of existing system: nExp. Our auditing time is approximately equal the scheme in existing system, it is only a difference of cExp. So we only need compare the communication cost of our auditing scheme with the work of existing system in experiments. The security level is chosen to be 80 bit, and |p| = |q| = 160. For simplicity, we also set k = 20, c = 300. All the results of experiments are represented as the average of 30 trials. As described performance analysis, the experimental results show that, compared with the auditing scheme in existing system, the communication cost of our auditing scheme are much light-weight than the scheme in existing system.

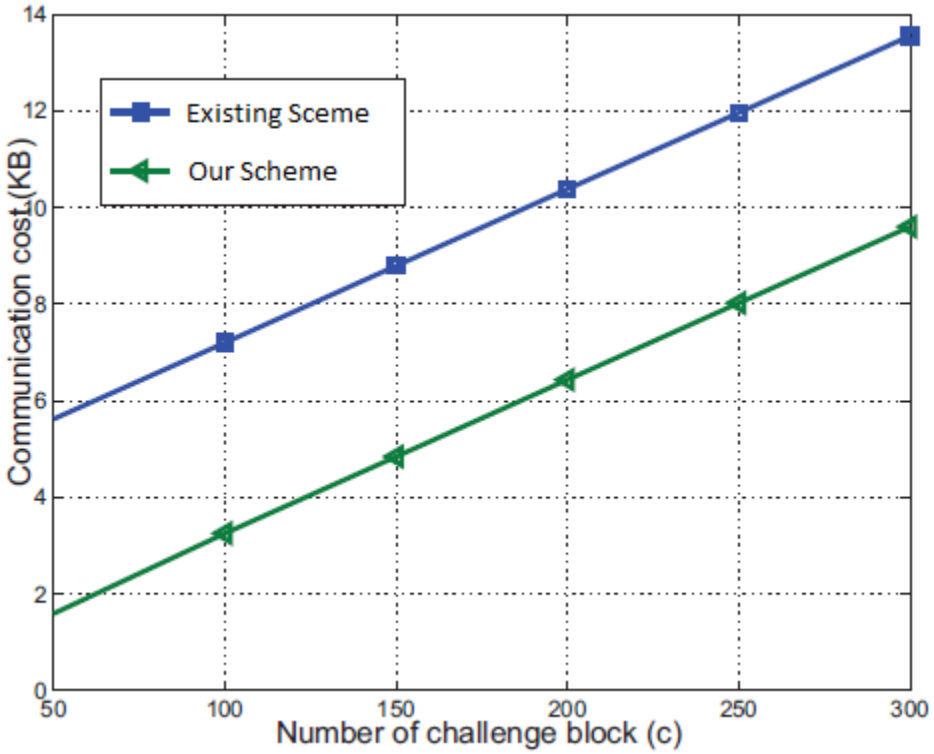

Fig 3: Comparison on the communication cost between our scheme and the Existing System

ACKNOWLEDGEMENT

This work is supported by the National Natural Sci-ence Foundation of China (No. 61272455), China 111 Project (No. B08038), Doctoral Fund of Ministry of Education of China (No. 20130203110004), Program for New Century Excellent Talents in University (No. NCET-13-0946), and the Fundamental Research Funds for the Central Universities (Nos. BDY151402 and JB142001-14).

7. CONCLUSION

The primitive of verifiable database with efficient updates is an important way to solve the problem of verifiable outsourcing of storage. We propose a scheme to realize efficient and secure data integrity auditing for share dynamic data with multi-user modification. The scheme vector commitment, Asymmetric Group Key

Agreement (AGKA) and group sig-natures with user revocation are adopt to achieve the data integrity auditing of remote data. Beside the public data auditing, the combining of the three primitive enable our scheme to outsource cipher text database to remote cloud and support secure group users revocation to shared dynamic data. We provide security analysis of our scheme, and it shows that our scheme provide data confidentiality for group users, and it is also secure against the collusion attack from the cloud storage server and revoked group users. Also, the performance analysis shows that, compared with its relevant schemes, our scheme is also efficient in different phases.

8. REFERENCES

[1] Amazon. (2007) Amazon simple storage service (amazon s3). Amazon. [Online]. Available: http://aws.amazon.com/s3/

[2] Google. (2005) Google drive. Google. [Online]. Available: http://drive.google.com/

[3] Dropbox. (2007) A file-storage and sharing service. Dropbox. [Online]. Available: http://www.dropbox.com/

[4] Mozy. (2007) An online, data, and computer backup software. EMC. [Online]. Available: http://www.dropbox.com/

[5] Bitcasa. (2011) Inifinite storage. Bitcasa. [Online]. Available: http://www.bitcasa.com/

[6] Memopal. (2007) Online backup. Memopal. [Online]. Available: http://www.memopal.com/

[7] M. A. et al., "Above the clouds: A berkeley view of cloud computing," *Tech. Rep. UCBEECS*, vol. 28, pp. 1–23, Feb. 2009.

[8] M. Rabin, "Efficient dispersal of information for security," *Journal of the ACM (JACM)*, vol. 36(2), pp. 335–348, Apr. 1989.

[9] J. G. et al. (2006) The expanding digital universe: A forecast of worldwide information growth through 2010. IDC. [Online]. Available: Whitepaper

[10] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at un-trusted stores," in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 598–609.

[11] A. Juels and B. S. Kaliski, "Pors: Proofs of retrievability for large files," in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 584–597.

[12] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: theory and implementation," in *Proc. of CCSW 2009*, llinois, USA, Nov. 2009, pp. 43–54.

[13] Y. Dodis, S. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in *Proc. of TCC 2009*, CA, USA, Mar. 2009, pp. 109–127.

[14] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Proofs of retrievability via hardness amplification," in *Proc. of ESORICS 2009*, Saint-Malo, France, Sep. 2009, pp. 355–370.

[15] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. of ACM CCS*, Illinois, USA, Nov. 2009, pp. 213–222.

[16] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proc. of IEEE INFOCOM 2010*, CA, USA, Mar. 2010, pp. 525–533.

[17] J. Yuan and S. Yu, "Proofs of retrievability with public verifiability and constant communication cost in cloud," in *Proc. of International Workshop on Security in Cloud Computing*, Hangzhou, China, May 2013, pp. 19–26.

[18] E. Shi, E. Stefanov, and C. Papamanthou, "Practical dynamic proofs of retrievability," in *Proc. of ACM CCS 2013*, Berlin, Germany, Nov. 2013, pp. 325–336.

[19] Cloud9. (2011) Your development environment, in the cloud. Cloud9. [Online]. Available: https://c9.io/

[20] Codeanywhere. (2011) Online code editor. Codeanywhere. [Online]. Available: https://codeanywhere.net/

[21] eXo Cloud IDE. (2002) Online code editor. Cloud IDE. [Online]. Available: https://codenvy.com/

[22] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," in *Proc. of IEEE CLOUD 2012*, Hawaii, USA, Jun. 2012, pp. 295–302.

[23] B. Wang, L. Baochun, and L. Hui, "Public auditing for shared data with efficient user revocation in the cloud," in *Proc. of IEEE INFOCOM 2013*, Turin, Italy, Apr. 2013, pp. 2904–2912.

[24] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification," in *Proc. of IEEE INFOCOM 2014*, Toronto, Canada, Apr. 2014, pp. 2121–2129.

[25] D. Catalano and D. Fiore, "Vector commitments and their applications," in *Public-Key Cryptography - PKC 2013*, Nara, Japan, Mar. 2013, pp. 55–72.