Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 9, August 2021: 8011-8018

Integrating Secure Network Coding Techniques to Store the Data Securely in Cloud with Data Dynamics

K Jaya Krishna[1], Dr. U. Mohan Srinivas[2], S. V. Jagadeeswari[3], P. Asha[4], O. Alekhya[5], V. Mamatha[6]
[1,2]Associate Professor, [3,4,5,6] PG Scholars
Department of MCA, QIS College of Engineering and Technology (Autonomous), Ongole

Abstract:
Cloud service providers offer stockpiling re-appropriating office to their customers. In a secure cloud stockpiling (SCS) convention, the respectability of the customer's information is kept up. In this work, we develop an openly evident secure cloud stockpiling convention in view of a secure organization coding (SNC) convention where the customer can refresh the rethought information on a case by case basis. To the best of our insight, our plan is the primary SNC-based SCS convention for dynamic information that is secure in the norm model and gives protection saving reviews in a freely undeniable setting. Besides, we examine, in subtleties, about the (im)possibility of giving an overall development of an productive SCS convention for dynamic information (DSCS convention) from a subjective SNC convention. Also, we change an existing DSCS conspire (DPDP I) to help privacypreserving reviews. We additionally contrast our DSCS convention and different SCS plans (counting the adjusted DPDP I conspire). At last, we sort out certain impediments of a SCS conspire built utilizing a SNC convention

1. Introduction

Cloud registration has become so popular due to its association with cloud computing. Despite that, many studies [1], [2] indicate that cloud service providers may have information mishaps (CSPs). This has prompted much interest in the problem of ensuring the security of cloud storage, as we previously referred to in our discussion of secure cloud storage (SCS). On the other side, others think that the organisation should do a better job of limiting trustworthiness checks. When there is a moderate changeover, intentional switching of codewords may cause problems with decoding, which leads to dissatisfaction at the endpoints. Trustworthiness of codewords is referenced as the secure coding problem of the secure organisation. Several scientists have investigated the pros and cons of private cloud storage and independent software writing. We've received some replies to the prior problem, for example [3], [4], and [5]. On the other hand, the final territory has been studied for more than a decade, for example [6] and [7]. Data storage on the cloud. Juels and Kaliski initially introduced this problem [3], and Ateniese et al. [4] followed. With these standards, two key components are in play: a client and a cloud storage provider. The customer is promised that the information would be kept safe on the cloud. The customer shows that they are serious about using the cloud to store sensitive information by joining a cloud capacity convention. Some key points serve as the basis for trustworthy information verification. With the cloud in its current state, the

client's data may be compromised due of design issues (equipment or programming). If there's a mistake, the cloud may pretend the client is mistaken. Second, cloud technology has a huge monetary incentive to erase information obtained by the customer. Ignoring some of the information makes a difference when it comes to saving money on the cloud. Third, a cloud might be hacked, too, which would allow the information to be manipulated. Fourth, a cloud may be malicious due to a variety of government pressures. If there was no accepted cloud storing standard, these cases could be disregarded and unremarked on by the cloud. An essential component of a safe cloud storage protocol is that clients may verify the information's authenticity without having to make use of the actual content. Other traditional methods of verification rely on hash and MAC codes, which also need the client to retain the information locally. Some conventions (for example, [5], [8], [9], and [10]) are freely verified, which means that anyone other than the client can verify the accuracy of the information; however, other conventions are secretly verified, meaning that only the client with the hidden key can verify the information's accuracy. The second section of the paper clarifies secure organisation code. Cai and Yeung [6] and Gkantsidis and Rodriguez [7] initially raised the problem. The organisational coding method is a systematic framework where information packets are sent through a network via a direct switch, as opposed to the more conventional store-and-forward system. Organizational limits may be extended with regard to multicast assignments by using encoding. Straightforward code, which is only sufficient to achieve the target limit [11], [12], is in reality quite suited to the application. This is especially useful in networks with members who are inclined to cooperate. However, this viewpoint has its own set of security issues. In the event that a packaged bundle is corrupted, this alteration will instantaneously propagate to the whole organisation because of the switch's ability to encode every received bundle, even those that are tainted. This attack is most often known as the contamination attack. Code word contamination may cause information disaster when information collectors try to figure out what the code words mean. Furthermore, since security is of concern in an organisation where data-driven businesses exist, beneficiaries and switches need to ensure if information is tainted when a new package is received. In general, an organization's secure coding problem is also a type of information integrity problem. Multiple mechanisms for ensuring data accuracy have been adopted, including cryptographic hash functions [13], MACs [14] and advanced marks [15], [16]. Every single code word in the company must be double-checked to ensure that it has not been tampered with. The packages in the company must be connected straightly by switches, and the new bundles must be validated. All existing solutions for company-level encryption use a few homomorphic properties of basic cryptographic algorithms. A store-and-forward directing method is replaced with a correspondence organisation coding technique. Each middle hub in an organisation coding (NC) convention will receive incoming bundles to make another package. Compared to just transferring a near package in its present condition, the organization's coding standards have the potential to provide greater throughput, productivity, and flexibility. But these standards are incapable of protecting against malicious and unpredictable attacks, such as when a harmful middleman injects contaminated packages into the network. Such packages are made more frequent downstream by the illegitimate bundles. In the worst-case scenario, the organization's goal hub is unable to parse the first document that is delivered to it. Coding standards with defences (for example, certain cryptographic natives) allow firms to discover the problem in question. SNC conventions use a source hub to examine each bundle before transmission. A little sticker is affixed to each package for parcel tracking. Chen et al. [14] examine

the link between static information stockpiling (SSCS) and secure organisation coding (SNC) in a brand-new task. SSCS conventions may be built using SNC conventions. However, with static information, customers (information proprietors) cannot change their information on the cloud server after transferring it. This setback means that an SSCS convention is absent in a lot of cloud programmes where customers have to update their information on a regular basis. While this may be an obvious way to update information in this scenario, downloading the whole record and transferring it to the server is very inefficient since it takes massive bandwidth for each change. This necessitates further studies on how to build a more successful (and more non-exclusive) dynamic information cloud (DISS) convention using a Single Negotiation Center (SNC) protocol.

2. Related Work

The goal of Juels and Kaliski [3] was to enable customers to verify the integrity of information transferred to the cloud. Clients should distribute some unusual data to the information (i.e., "sentinel" [3]) at random places. Despite the fact that the confirmation data is unclear, generated randomly, and is unrelated to the information, the cloud has no clue where they are located. To validate the client, the cloud will reply with either the unusual validation information or the normal information the client has entered. While this method has its merits, it has one major downside: the absolute number of sentinels is restricted, which means the review process must be repeated a limited number of times. Ateniese et al. [4] suggested the potential of an ownership method that uses homomorphic verification information for proving verifiability. It is normally feasible to do calculations on blocks of information, in order to make another authenticator. At that moment, the client may ask the cloud for a computation of the random cloud blocks, as well as a verification of the registered result. If the confirmation is correct, the cloud does not corrupt the client's information. In order to reduce the response and inquiry time, two techniques based on pseudorandom capabilities and a combining mark with formal security verifications were suggested [8]. The team made up of outsiders evaluates the privacy-preserving outsider. A similarly blended alternative convention is suggested afterwards [10]. In consideration of a brand-new convention that endeavours to reduce communication fees, a convention is suggested [9]. Moreover, some interesting work is based on hash numbers in a hypothetical framework [17]. All of the aforementioned strategies are used in a creative manner. Moreover, we provide a precise, non-exclusive approach to setup.

3. Secure Cloud Storage

3.1 System Model, Threat Model, and Design Goals

Our solution shows a secure cloud storage paradigm in Fig. 1. The two halves of this system are the client and the cloud. A customer may be a person, a business or an organisation that uses a PC or a mobile phone, etc.; a customer could be a CSP, such as Amazon S3, Dropbox, Google Drive, etc. To begin, the client retrieves its data from the cloud. At a later date, the customer checks the re-appropriated data's veracity and credibility. It is then possible for the client to ascertain whether or not the cloud verification returned a valid result, meaning that the data is preserved, or that the proof that the data has been altered is obtained, which may prompt additional action (which is outside of our scope), such as legal action or data retrieval. Our study in this article follows the approach of the

previous works [3], [4], and we posit that the cloud is potentially dangerous. As far as we know, the exchange between the client and the cloud can be authenticated using conventional methods. Therefore, we can focus on the client and the cloud without having to deal with communications. The creation of a cloud storage system that allows clients to audit data authenticity is necessary:

• Correct. It is possible for the cloud to always prove to clients that the data is intact, even if the cloud keeps a copy of every re-appropriated file.

• Secure. Even if the cloud is trying to cover for the customer in the audit, the client can detect with high probability in the event that their data is destroyed.

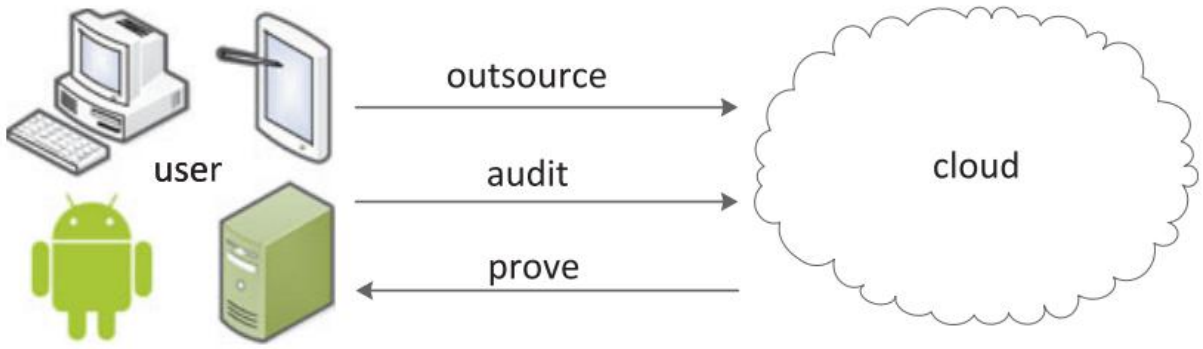

Fig. 1. A secure cloud storage system.

• Efficient. It is imperative that the client and the cloud both maintain as minimal a compute, storage, and communication cost as possible.

3.2 High-Level Protocol Specification

We now model a framework for securely storing data in the cloud. The client's server has to have some certain data on it that is processed to a particular security level in order to validate whether the cloud is telling the truth to an audit enquiry. Label K as restricted data, a number represents the security level, and F represents the client's data. K is used by the customer to overcome F. At this moment, the data including authentication information, represented by F0, is transferred to the cloud. When the cloud gets an audit inquiry q from the client, it generates a verification G that confirms the data is intact using the F0 put away data. Regardless of whether G is legitimate, the client checks at that moment. Let the customer know his verification status by marking it with d. Furthermore, an encrypted cloud storage protocol includes five very effective algorithms. There is authentication information F with certain data produced in the cloud, which must be verified and subsequently transported out of the cloud.

3.3 Understanding Security

This chapter offers a clear definition of the secure cloud storage protocol by dissecting its real-world application bit by bit. Characterizing the concept of security can help you comprehend the security. Before we can proceed, we need to assess the harmful capabilities of a cloud. The database is already loaded for the project. The cloud may observe a huge number of audits in addition to its positive responses. Furthermore, it's fair to assume that the cloud can ascertain if the client accepts a proof of

reaction. Since clients may choose to sue the cloud or pursue some other course of action if they reject the proof, they are less likely to do so if they accept the verification. Another key problem is the cloud's capacity to audit and verify information. We define a malicious cloud that can view exponentially many such queries and answers. This serves the audit needs of a customer.

4. DPDP I: A Dynamic Provable Data

Schemes of Possession Erway et al. [18, 17] offer two proven data possession systems, productive and totally dynamic: DPDP I (based on the authentication of the ranks) and DPDP II; (based on rank-based RSA trees). In this analysis, we look just at the DPDP I strategy.

4.1 Blockless Verification in DPDP I

A public key generation method called KeyGen is used to create a key pk with two big primes and a Z N component with a huge request. Let ˜m blocks b1, b2, . . . , bm˜ constitute the initial data record. The customer does the following for each tag T (b) = g b mod N: Nowadays, customers use skip lists, which are verified by ranks, and they assemble the list on the tags of the squares, uploading the data, tags, and skip list to the cloud server. Like the prior investigations, the inclusion, deletion, and alteration steps follow a similar procedure. The DPDP I protocol does not have a secret key. While Erway et al. attempt to avoid making sweeping statements about the public verifiability of the DPDP I scheme, we may make the metadata dM˜ of the forward-thinking skip list and the value ˜m accessible to everyone by simply making them public. The verifier selects a random subset of l-component I, which is then used to create a challenge set where each element is a random value. The challenge set Q is sent to the server by the verifier.

4.2 Modified DPDP I to Make Audits Privacy Preserving

When an external auditor (TPA) conducts a privacy-safeguarding audit on the dynamic data in the secure cloud storage, they are unable to discern the true content of the data. Let us determine if the DPDP I programme that this facility received is working. Just like in the original design [18], the server transmits the aggregated square νibi, which is made up of the vectors v i∈I , to the verifier (or TPA), where l is the total number of dimensions. In the current state of affairs, a TPA may work out the bi values by solving a system of linear equations. Consequently, the initial audits are not designed to safeguard privacy. Nevertheless, it's not difficult to design these checks to ensure privacy protection. We make adjustments to audit systems like such. The verifier again sends the server the challenge set Q = I νi)}i∈I. The server processes a square obtained by aggregating the rows of P = I ˆbi. The server has a selection process that randomly chooses a value r, then processes B 0 = B + r and R = g r mod N. The server transmits bi to the verifier, as well as a reset value, B 0 , and the verifications Π(i) to the verifier.

5. Performance Analysis

Now, we'll focus on the efficiency of our DSCS protocol and see how it stands in comparison to competing SCS schemes with proven data ownership guarantees. While we understand certain drawbacks of a SCS solution that uses an SNC (for static or dynamic data) compared to the DPDP I solution, we appreciate its advantages.

Efficiency

It is mostly the time it takes to do exponentiations (modulo N) that dominates the computational cost of our DSCS protocol. The client must do a multi-exponentiation and compute the e-th base of the result to produce the value x in an authentication tag for each vector (in the algorithm Outsource). To get the value of x, the server uses two multi-exponentiations. The Verify algorithm must be implemented to do multi- and single- exponentiations to verify a proof. Since the characteristics of a skip list call for it, because of the properties of a skip list, verification Π (as linked to the rank-based authenticated skip list) has a size of O(log m) with high probability.

a comparison of the five PDP plans. To support our PDP guarantee, we'll compare our DSCS protocol with various PDP schemes available in the literature. At this point, we consider some of the constraints of our DSCS protocol, particularly in comparison to DPDP I, since both are secure, they manage dynamic data, and provide public verifiability in the standard model. In the DSCS protocol, audit information is safeguarded from unwanted intruders by the use of privacy measures.

6. Experimental Results for DSCS II

The additional storage cost (storage overhead at the server S) for DSCS II accounts for the authentication tags only. The storage overhead is reported in Table . After the initial outsourcing of the data file, the client C and the server S need to communicate with each other either during an audit or during an append. The communication cost during an audit for Q data blocks includes the size of the aggregated tag (the size of the single aggregated block is not reported here). The communication costs for an audit and an append are reported in Table. This summarizes the computation cost for DSCS II incurred during the initial outsourcing, an audit and an append. We note that the time needed to generate a tag is much more expensive (compared to DSCS I) due to the costly operations in bilinear groups implemented using the PBC library.

Storage overhead for the server S and communication cost in DSCS II

File size (MB)	Storage cost for tags (KB)	Storage overhead	$\|Q\|$	Communication cost (B)	
				Audit	Append
1	0.39	0.04%	2	131	131
10	2.751	0.03%	10	131	131
50	13.23	0.03%	10	131	131
200	52.53	0.03%	112	131	131
500	131.00	0.03%	112	131	131

Computation cost for the client C and the server S in DSCS II

File size (MB)	Outsource (sec)	$\|Q\|$	Challenge (msec)	Prove (sec)	Verify (sec)	Append (sec)	
	C		C	S	C	C	S
1	154.05	2	0.09	0.12	0.02	65.75	0.02
10	1459.06	10	0.32	0.75	0.03	65.49	0.24
50	7212.21	10	0.47	0.79	0.05	65.45	1.15
200	28600.80	112	3.78	12.85	0.37	65.34	8.90
500	96001.83	112	4.43	13.49	0.39	65.51	20.40

7. Conclusion

Our research has outlined a DSCS method that is based on an SNC protocol. According to our understanding, this is the first safe, publicly verifiable, and privacy-preserving SNC-based DSCS protocol. In addition, we have explored the characteristics of an SNC protocol that allows us to create an efficient DSCS protocol that utilises this SNC protocol. A previous DSCS system (DPDP I [18]) has been updated to be privacy-preserving. Our DSCS architecture was examined, and we compared it with other secure cloud storage methods to determine that it provides proof of data ownership. It has now been determined that SNC-based secure cloud storage protocols have some restrictions. Nonetheless, some of these constraints are in fact a consequence of the underlying SNC protocols that are being used. A new SNC protocol with improved efficiency will offer us a better DSCS protocol.

References

[1] S. Agrawal and D. Boneh. Homomorphic MACs: MAC-based integrity for network coding. In Applied Cryptography and Network Security - ACNS 2009, pages 292–305, 2009.

[2] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. IEEE Transactions on Information Theory, 46(4):1204–1216, 2000.

[3] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, and D. X. Song. Provable data possession at untrusted stores. In ACM Conference on Computer and Communications Security, CCS 2007, pages 598–609, 2007.

[4] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik. Scalable and efficient provable data possession. In International Conference on Security and Privacy in Communication Networks, SECURECOMM 2008, pages 9:1–9:10, 2008.

[5] N. Attrapadung and B. Libert. Homomorphic network coding signatures in the standard model. In Public Key Cryptography - PKC 2011, pages 17–34, 2011.

[6] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In ACM Conference on Computer and Communications Security, CCS 1993, pages 62–73, 1993.

[7] D. Boneh, D. M. Freeman, J. Katz, and B. Waters. Signing a linear subspace: Signature schemes for network coding. In Public Key Cryptography - PKC 2009, pages 68–87, 2009.

[8] K. D. Bowers, A. Juels, and A. Oprea. HAIL: A high-availability and integrity layer for cloud storage. In ACM Conference on Computer and Communications Security, CCS 2009, pages 187–198, 2009.

[9] K. D. Bowers, A. Juels, and A. Oprea. Proofs of retrievability: Theory and implementation. In ACM Cloud Computing Security Workshop, CCSW 2009, pages 43–54, 2009.

[10] D. Cash, A. Kup¸c ¨ u, and D. Wichs. Dynamic proofs of ¨ retrievability via oblivious RAM. In Advances in Cryptology - EUROCRYPT 2013, pages 279–295, 117 2013.

[11] D. Catalano, D. Fiore, and B. Warinschi. Efficient network coding signatures in the standard model. In Public Key Cryptography - PKC 2012, pages 680–696, 2012.

[12] N. Chandran, B. Kanukurthi, and R. Ostrovsky. Locally updatable and locally decodable codes. In Theory of Cryptography Conference, TCC 2014, pages 489–514, 2014.

[13] D. X. Charles, K. Jain, and K. E. Lauter. Signatures for network coding. International Journal of Information and Coding Theory, 1(1):3–14, 2009.

[14] F. Chen, T. Xiang, Y. Yang, and S. S. M. Chow. Secure cloud storage meets with secure network coding. In IEEE Conference on Computer Communications, INFOCOM 2014, pages 673–681, 2014.
[15] R. Curtmola, O. Khan, R. C. Burns, and G. Ateniese. MR-PDP: multiple-replica provable data possession. In IEEE International Conference on Distributed Computing Systems - ICDCS 2008, pages 411–420, 2008.
[16] Y. Dodis, S. P. Vadhan, and D. Wichs. Proofs of retrievability via hardness amplification. In Theory of Cryptography Conference, TCC 2009, pages 109–127, 2009.
[17] C. C. Erway, A. Kup¸c ¨ u, C. Papamanthou, and ¨ R. Tamassia. Dynamic provable data possession. In ACM Conference on Computer and Communications Security, CCS 2009, pages 213–222, 2009.
[18] C. C. Erway, A. Kup¸c ¨ u, C. Papamanthou, and ¨ R. Tamassia. Dynamic provable data possession. ACM Transactions on Information and System Security, 17(4):15, 2015.
[19] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin. Secure network coding over the integers. In Public Key Cryptography - PKC 2010, pages 142–160, 2010.
[20] M. T. Goodrich, R. Tamassia, and A. Schwerin. Implementation of an authenticated dictionary with skip lists and commutative hashing. In DARPA Information Survivability Conference and Exposition (DISCEX) II, pages 68–82, 2001.
[21] T. Ho, R. Koetter, M. M´edard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In IEEE International Symposium on Information Theory - ISIT 2003, page 442, 2003.
[22] T. Ho, M. M´edard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. IEEE Transactions on Information Theory, 52(10):4413–4430, 2006.