Journal of Engineering Sciences

Vol 12, Issue 06, JUNE /2021
ISSN NO:0377-9254

Secure Network Coding Performances Using Public Provable Secure Data Dynamics With Cloud Storage

MEHATAZ MULLA, HARITHA LAKSHMI BOREDDY, SASIKALA SAPPIDI, K AMARENDRANATH, Dr.G RAJESH CHANDRA
DEPT OF CSE
SVR ENGINEERING COLLEGE, NANDYAL

ABSTRACT

In the age of cloud computing, cloud users with limited storage can outsource their data to remote servers. These servers, in lieu of monetary benefits, offer retrievability of their clients' data at any point of time. Secure cloud storage protocols enable a client to check integrity of outsourced data. In this work, we explore the possibility of constructing a secure cloud storage for dynamic data by leveraging the algorithms involved in secure network coding. We show that some of the secure network coding schemes can be used to construct efficient secure cloud storage protocols for dynamic data, and we construct such a protocol (DSCS I) based on a secure network coding protocol. To the best of our knowledge, DSCS I is the first secure cloud storage protocol for dynamic data constructed using secure network coding techniques which is secure in the standard model. Although generic dynamic data support arbitrary insertions, deletions and modifications, append-only data find numerous applications in the real world. We construct another secure cloud storage protocol (DSCS II) specific to append-only data — that overcomes some limitations of DSCS I. Finally, we provide prototype implementations for DSCS I and DSCS II in order to evaluate their performance.

I. INTRODUCTION:

WITH the advent of cloud computing, cloud servers offer to their clients (cloud users) various services that include delegation of huge amount of computation and outsourcing large amount of data. For example, a client having a smart phone with a low-performance processor or limited storage cannot accomplish heavy computation or store large volume of data. Under such circumstances, she can delegate her computation/storage to the cloud server. In case of storage outsourcing, the cloud server stores massive data on behalf of its clients (data owners). However, a malicious cloud server can delete some of the client's data (that are accessed infrequently) to save some space. Secure cloud storage protocols (two-party protocols between the client and the server) provide a mechanism to detect if the server stores the client's data un tampered. Based on the nature of the outsourced data, these protocols are classified as: secure cloud storage protocols for static data (SSCS) [2], [3], [4] and for dynamic data (DSCS) [5], [6], [7], [8]. For static data, the client cannot change her data after the initial outsourcing (e.g., backup/archival data). Dynamic data are more generic in that the client can modify her data as often as needed. In secure cloud storage protocols, the client can audit the outsourced data without accessing the whole data file, and still be able to detect unwanted changes in data done by a malicious server. During an audit, the client sends a random challenge to the server which produces proofs of storage (computed on the stored data) corresponding to that challenge. Secure cloud storage protocols are publicly verifiable if an audit can be performed by any third party auditor (TPA) using public parameters; or privately verifiable if an auditor needs some secret information of the client. The entities involved in a secure cloud storage protocol and the interaction among them are shown in Figure 1. In a network coding protocol [9], [10], each intermediate node (except sender/receiver nodes) on a network path combines incoming packets to output another packet. These protocols enjoy higher throughput, efficiency and scalability than the store-and-forward routing, but they are prone to pollution attacks by malicious intermediate nodes injecting invalid packets. These packets produce more such packets downstream, and the receiver might not finally decode the file sent by the sender node. Secure network coding (SNC) protocols use cryptographic

techniques to prevent these attacks: the sender authenticates each packet by attaching a small tag to it. These authentication tags are generated using homomorphic message authentication codes (MACs) [11] or homomorphic signatures [12], [13], [14], [15]. Due to homomorphic property, an intermediate node can combine incoming packets (and their tags) into a packet and its tag.

In this work, we look at the problem of constructing a secure cloud storage protocol for dynamic data (DSCS) construct an efficient DSCS protocol using an SNC protocol. In a previous work, Chen et al. [16] reveal a relationship between secure cloud storage and secure network coding. In particular, they show that one can exploit some of the algorithms involved in an SNC protocol in order to construct a secure cloud storage protocol for static data. However, their construction does not handle dynamic data — that makes it insufficient in many applications where a client needs to update (insert, delete or modify) the remote data efficiently. Further investigations are needed towards an efficient DSCS construction using a secure network coding (SNC) protocol. Network coding techniques have been used to construct distributed storage systems [17], [18] where the client's data are disseminated across multiple servers. However, they primarily aim to reduce the repair bandwidth when some of the servers fail. On the other hand, we explore whether we can exploit the algorithms involved in an SNC protocol to construct an efficient and secure cloud storage protocol for dynamic data (for a single storage server). Although dynamic data are generic in the sense that they support arbitrary update (insertion, deletion and modification) operations, append-only data (where new data corresponding to a data file are inserted only at the end of the file) find numerous applications as well. These applications primarily maintain archival as well as current data by appending the current data to the existing datasets. Examples of append-only data include data obtained from CCTV cameras, ledgers containing monetary transactions, medical history of patients, data stored at append-only databases, and so on. Append-only data are also useful for maintaining other log structures (e.g., certificates are stored using append-only log structures in certificate transparency schemes [39]). In many of such applications, the data owner requires a cloud server to store the bulk data in an un tampered and retrievable fashion with append being the only permissible update. Although secure cloud storage schemes for generic dynamic data also work for append-only data, a more efficient solution (specific to append-only data files)
would be helpful in this scenario.

II. EXISTING SYSTEM:

Cloud Computing has been envisioned as the next-generation architecture of IT Enterprise. It moves the application software and databases to the centralized large data centers, where the management of the data and services may not be fully trustworthy. This unique paradigm brings about many new security challenges, which have not been well understood. Existing work studies the problem of ensuring the integrity of data storage in Cloud Computing. In particular, we consider the task of allowing a third party auditor (TPA), on behalf of the cloud client, to verify the integrity of the dynamic data stored in the cloud. The introduction of TPA eliminates the involvement of the client through the auditing of whether his data stored in the cloud are indeed intact, which can be important in achieving economies of scale for Cloud Computing.

The support for data dynamics via the most general forms of data operation, such as block modification, insertion, and deletion, is also a significant step toward practicality, since services in Cloud Computing are not limited to archive or backup data only. While prior works on ensuring remote data integrity often lacks the support of either public audit ability or dynamic data operations, this paper achieves both. The existing system first identifies the difficulties and potential security problems of direct extensions with fully dynamic data updates from prior works and then shows how to construct an elegant verification scheme for the seamless integration of these two salient features in our protocol design.

In particular, to achieve efficient data dynamics, we improve the existing proof of

storage models by manipulating the classic Merkle Hash Tree construction for block tag authentication. To support efficient handling of multiple auditing tasks, we further explore the technique of bilinear aggregate signature to extend our main result into a multiuser setting, where TPA can perform multiple auditing tasks simultaneously. Extensive security and performance.

Disadvantages

- In the existing work, the system is less effective due to absent Data integrity proof to verify the data from the cloud server.
- The system is less security due to absence of system which is publicly verifiable DSCSprotocol (DSCS I) from the SNC protocol proposed which is secure in the standard model..

III. PROPOSED SYSTEM:

- Network coding techniques have been used to construct distributed storage systems [17], [18] where the client's data are disseminated across multiple servers. However, they primarily aim to reduce the repair bandwidth when some of the servers fail. On the other hand, we explore whether we can exploit the algorithms involved in an SNC protocol to construct an efficient and secure cloud storage protocol for dynamic data (for a single storage server).
- Although dynamic data are generic in the sense that they support arbitrary update (insertion, deletion and modification) operations, append-only data (where new data corresponding to a data file are inserted only at the end of the file) find numerous applications as well. These applications primarily maintain archival as well as current data by appending the current data to the existing datasets. Examples of append-only data include data obtained from CCTV cameras, ledgers containing monetary transactions, medical history of patients, data stored at append-only databases, and so on.
- Append-only data are also useful for maintaining other log structures (e.g., certificates are stored using append-only log structures in certificate transparency schemes [39]). In many of such applications, the data owner requires a cloud server to store the bulk data in an un tampered and retrievable fashion with append being the only permissible update. Although secure cloud storage schemes for generic dynamic data also work for append-only data, a more efficient solution (specific to append-only data files) would be helpful in this scenario
- The system explores the possibility of providing a generic construction of a DSCS protocol from any SNC protocol. We discuss the challenges for a generic construction in details and identify some SNC protocols suitable for constructing efficient DSCS protocols.
- The proposed system constructs a publicly verifiable DSCS protocol (DSCS I) from an SNC protocol [15]. DSCS I handles dynamic data, i.e., a client can efficiently perform updates (insertion, deletion and modification) on the outsourced data. We discuss the (asymptotic) performance and certain limitations of DSCS I.
- The proposed s system provides the formal security definition of a DSCS protocol and prove the security of DSCS I. _ As append-only data are a special case of generic dynamic data, we can use DSCS I (which is based on [15]) for append-only data. However, we identify some SNC protocols that are not suitable for building a secure cloud storage for generic dynamic data, but efficient secure cloud storage protocols for append only data can be constructed from them. We construct such a publicly verifiable secure cloud storage protocol (DSCS II) for append-only data by using an SNC protocol proposed by Boneh et al. [13].

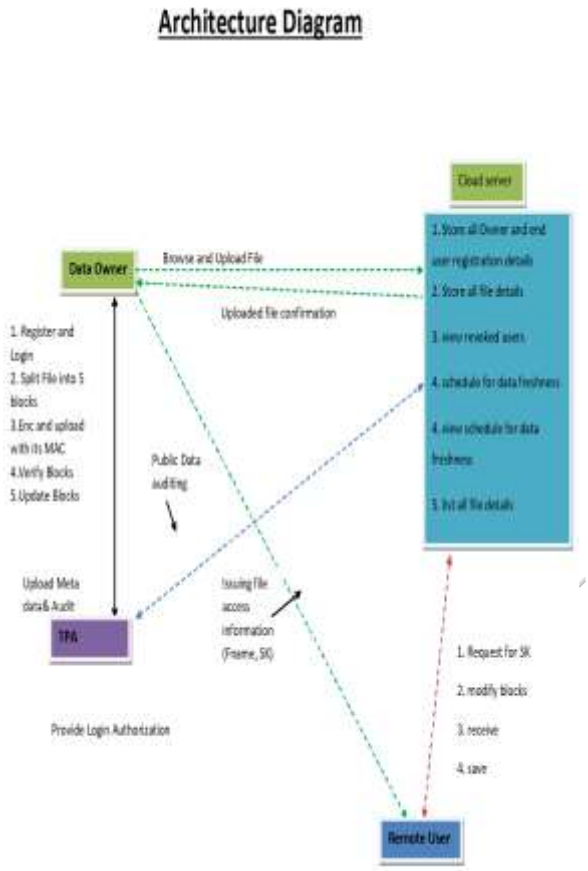
Advantages

- The system is more effective due to presence of Provable

Data Possession (PDP) techniques.

- The system is more accurate due to presence of Authenticated Data Structures Used in DSCS.

IV. SYSTEM ARCHITECTURE:



V. MODULES:

1. **Data Owner**

In this module, the data owner uploads their data with its File Blocks block in the cloud server. For the security purpose the data owner encrypts the data File Blocks and then store in the cloud. The data owner can change the policy over data File Blocks by updating the expiration time. The Data owner can have capable of manipulating the encrypted data File Blocks. And the data owner can set the access privilege to the encrypted data File Blocks.

2. **Cloud Server**

The cloud service provider manages a cloud to provide data storage service. Data owners encrypt their data File Blocks and store them in the cloud for sharing with data consumers. To access the shared data File Blocks, data consumers download encrypted data File Blocks of their interest from the cloud and then decrypt them.

3.Third Party Auditor

Third party auditor (TPA), who has capabilities to manage or monitor the outsourced data under the delegation of data owner, who has expertise and capabilities that a common user does not have, for periodically auditing the outsourced data. This audit service is significantly important for digital forensics and credibility in clouds and setting time period to update the old secret keys to new secret keys.

4. End User

The Cloud User who has a large amount of data to be stored in cloud and have the permissions to access and manipulate stored data and performs the following operations such as Searches for File Blocks based on Content's keyword, Requests for File Blocks, Request File Blocks for downloading with current sec key for the corresponding File Blocks from the cloud and dec, download

VI. SYSTEM SPECIFICATION:

H/W System Configuration:-

- Processor - Pentium –IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

Software Requirements:

- Operating System - Windows XP
- Coding Language - Java/J2EE(JSP,Servlet)
- Front End - J2EE

Back End - MySQL

VII. CONCLUSION:

In this work, we have proposed a secure cloud storage protocol for dynamic data (DSCS I) based on a secure network coding (SNC) protocol. To the best of our knowledge, this is the first SNC-based DSCS protocol that is secure in the standard model and enjoys public verifiability. We have discussed some challenges while constructing an efficient DSCS protocol from an SNC protocol. We have also identified some limitations of an SNC-based secure cloud storage protocol for dynamic data. However, some of these limitations follow from the underlying SNC protocol used. A more efficient SNC protocol can give us a DSCS protocol with better efficiency.We have also identified certain SNC protocols suitable for append-only data and constructed an efficient DSCS protocol (DSCS II) for appendonly data. We have shown that DSCS II overcomes some limitations of DSCS I. Finally, we have provided prototype implementations of DSCS I and DSCS II in order to show their practicality and compared the performance of DSCS I with that of an SNC-based secure cloud storage for static data and that of DPDP I.

VIII. REFERENCES:

[1] B. Sengupta and S. Ruj, "Publicly verifiable secure cloud storage
for dynamic data using secure network coding," in ACM Asia
Conference on Computer and Communications Security, 2016, pp. 107–
118.

[2] 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, 2007, pp. 598–609.

[3] A. Juels and B. S. Kaliski, "PORs: Proofs of retrievability for
large files," in ACM Conference on Computer and Communications
Security, 2007, pp. 584–597.

[4] H. Shacham and B. Waters, "Compact proofs of retrievability,"
Journal of Cryptology, vol. 26, no. 3, pp. 442–483, 2013.

[5] C. C. Erway, A. K¨upc¸ ¨ u, C. Papamanthou, and R. Tamassia,
"Dynamic provable data possession," ACM Transactions on
Information and System Security, vol. 17, no. 4, pp. 15:1–15:29, 2015.

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

[7] D. Cash, A. K¨upc¸ ¨ u, and D. Wichs, "Dynamic proofs of
retrievability via oblivious RAM," in EUROCRYPT, 2013, pp.
279–295.

[8] E. Shi, E. Stefanov, and C. Papamanthou, "Practical dynamic
proofs of retrievability," in ACM Conference on Computer and
Communications Security, 2013, pp. 325–336.

[9] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network
information flow," IEEE Transactions on Information Theory, vol. 46,
no. 4, pp. 1204–1216, 2000.

[10] S. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," IEEE
Transactions on Information Theory, vol. 49, no. 2, pp. 371–381, 2003.

[11] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based

integrity for network coding," in International Conference on
Applied Cryptography and Network Security, 2009, pp. 292–305.
[12] D. X. Charles, K. Jain, and K. E. Lauter, "Signatures for network
coding," International Journal of Information and Coding Theory,
vol. 1, no. 1, pp. 3–14, 2009.
[13] D. Boneh, D. M. Freeman, J. Katz, and B. Waters, "Signing a linear
subspace: Signature schemes for network coding," in International
Conference on Practice and Theory in Public Key Cryptography, 2009,
pp. 68–87.
[14] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin, "Secure network
coding over the integers," in International Conference on Practice
and Theory in Public Key Cryptography, 2010, pp. 142–160.
[15] D. Catalano, D. Fiore, and B. Warinschi, "Efficient network coding
signatures in the standard model," in International Conference on
Practice and Theory in Public Key Cryptography, 2012, pp. 680–696.
[16] F. Chen, T. Xiang, Y. Yang, and S. S. M. Chow, "Secure cloud
storage meets with secure network coding," in IEEE International
Conference on Computer Communications, 2014, pp. 673–681.
[17] A. G. Dimakis, B. Godfrey, Y. Wu, M. J. Wainwright, and
K. Ramchandran, "Network coding for distributed storage
systems," IEEE Transactions on Information Theory, vol. 56, no. 9,
pp. 4539–4551, 2010.
[18] K. Omote and T. T. Phuong, "DD-POR: Dynamic operations and
direct repair in network coding-based proof of retrievability,"
in International Computing and Combinatorics Conference, 2015, pp.
713–730.
[19] "Secure cloud storage," 2018. https://github.com/akankshadixit/
Secure Cloud Storage
[20] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for
cryptographers," Discrete Applied Mathematics, vol. 156, no. 16, pp.
3113–3121, Sep. 2008.
[21] 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, vol. 52, no. 10,
pp. 4413–4430, 2006.
[22] 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, 2003, p. 442.
[23] 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, 2008, p. 9.
[24] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou,
"Privacy-preserving public auditing for secure cloud storage,"
IEEE Transactions on Computers, vol. 62, no. 2, pp. 362–375, 2013.
[25] F. J. MacWilliams and N. J. A. Sloane, The Theory of Error-Correcting
Codes. North-Holland Publishing Company, 1977.
[26] I. S. Reed and G. Solomon, "Polynomial codes over certain finite
fields," Journal of the Society for Industrial and Applied Mathematics,
vol. 8, no. 2, pp. 300–304, 1960.
[27] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability:
Theory and implementation," in ACM Cloud Computing Security
Workshop, 2009, pp. 43–54.
[28] Y. Dodis, S. P. Vadhan, and D. Wichs, "Proofs of retrievability

via hardness amplification," in Theory of Cryptography Conference,
2009, pp. 109–127.
[29] E. Stefanov, M. van Dijk, A. Juels, and A. Oprea, "Iris: A scalable
cloud file system with efficient integrity checks," in Annual
Computer Security Applications Conference, 2012, pp. 229–238.
[30] N. Chandran, B. Kanukurthi, and R. Ostrovsky, "Locally
updatable and locally decodable codes," in Theory of Cryptography
Conference, 2014, pp. 489–514.