ISSN: 2278-4632
Vol-12 Issue-01 No.01: 2022

DYNAMIC AND SECURE CLOUD STORAGE USING NETWORK CODING TECHNIQUES

#1EJJAGIRI SUSHMITHA, M.Tech Student, Dept of CSE,
#2Dr.CHANDRAMOULI NARSINGOJU, Associate Professor & HOD, Dept of CSE,
#3Dr. GULAB SINGH, Associate Professor, Dept of CSE,
VAAGESWARI COLLEGE OF ENGINEERING, KARIMNAGAR, TS.

ABSTRACT—
In this study, it was discovered for the first time that there is a connection between cloud storage and secure network coding techniques. The concept of secure cloud storage has just lately been introduced in the context of network coding and data storage, and it is still in its early stages. Even though the two domains are vastly different in nature and have been investigated independently, we demonstrate how to design a secure cloud storage protocol using any secure network coding approach available. These days, cloud storage methods may be designed in a methodical manner, thanks to technological advancements. The security of our architecture has been determined by an evaluation methodology that takes into account real-world cloud storage usage. Also included in the scope of this research are two specialised cloud storage systems that are based on two current secure network coding protocols that have been discussed in full elsewhere. As part of this architecture, we also develop the world's first publicly verifiable cloud storage method, which is a first for the industry as a whole. As an added bonus, we are now working on implementing user anonymity as well as third-party public audits into the overall structure that has been presented. Finally, we create a working prototype of the new protocol and conduct a performance evaluation to determine its effectiveness and efficiency in real-world situations. The protocol's effectiveness has been demonstrated in experimental investigations.

***Index Terms*—** A few examples of data encryption techniques are dynamic data, append-only data, public verifiability, Cloud storage audits and security. Other examples include network coding and security.

---

1. INTRODUCTION

As a result of the growing usage of cloud computing, cloud storage is becoming increasingly popular among consumers. Recently conducted research have showed that cloud storage services are prone to data loss and corruption, despite these precautions (CSPs). Therefore, secure cloud storage (SCS), a term we developed to describe the difficulties of validating the integrity of data saved in the cloud, has attracted the interest of a large number of people. The integrity checks performed on network coding, which has been proposed as a method to increase network capacity, have been found to be insufficient. When an intermediary router actively tampers with codewords, it is probable that decoding problems will emerge. It is referred to as the secure network coding problem because it is concerned with the process of determining the integrity of a codeword while it is transmitted via a network connection. Different research on safe cloud storage, as well as secure network coding, have been undertaken by scholars from a number of different institutions and disciplines. These are instances of solutions that have only recently been given to the preceding problem, as indicated by the numbers 3, 4, and 5. According to [6] and [7], the latter area has been the subject of research for at least a decade or longer.

Secure cloud storage.

Ateniese et al. [4] were among the first to propose solutions to this problem, following Juels and Kaliski [3]. The user and the cloud storage service are the two most significant participants in these protocols, with the user coming in second. Using cloud-based storage services, a user can put their data in the hands of a third party. When data is stored in a secure cloud storage environment, the accuracy of the information can be checked by the user. There are a multitude of reasons that contribute to the necessity for data integrity testing. Users' data may be lost in the event of a system failure due to the cloud's weak control, to name a few concerns (hardware or software). The cloud may give the user a lie in order to keep the nature of the disaster hidden from the user's view. It is in the cloud's commercial interest to dispose of data as rapidly as possible if you have an excessive volume of data that isn't being used by the cloud. The cloud is able to save money by ignoring a portion of the data collected. As a third option, hacking attacks on cloud-based data can be used to gain access to and manipulate the data stored on the cloud. A cloud could also be malevolent as a result of different government pressures, such as censorship, on the Internet, which could result in a cloud being malicious. In order to prevent these instances from going undetected, it is required to establish a secure cloud storage mechanism. It is feasible to verify the integrity of data stored using cloud storage systems without having to physically access the data. Most traditional systems, such as those that rely on hashes, message authentication codes, and digital signatures, require that data be stored on a local computer. Those protocols that can be verified by anyone other than the user are known as publicly verifiable protocols (for example, [5], [8], [9], [10]), whereas those that can only be verified by the user are known as privately verifiable protocols (for example, [5], [8], [9], and [10]).

Secure network coding.

Gkantsidis and Rodriguez [7] were the first to offer the concept of this problem, which was previously proposed by Gkantsidis [6]. Instead of the usual store and forward mode of communication, a network router employs network coding to send out encoded data packets that are a function of the data packets it has previously received. Multicast activities might benefit from the additional network capacity provided by encoding in order to increase their overall performance. It has been demonstrated in [11] and [12] that linear coding is sufficient for increasing capacity. This will be especially beneficial to cooperative networks, which will benefit from it. A substantial number of security risks are associated with this paradigm, on the other hand. Encoding is used to protect each and every packet received by a router, including those that have been tampered with. Therefore, if an encoded packet is unlawfully modified, the modification will spread rapidly throughout the whole network. Among the many variations on this type of attack is the pollution attack. The data recipient may experience data loss as a result of the tainted code words being used during the decoding procedure on their end. When security is a big concern, data receivers and routers must check each packet to detect if it has been contaminated with malware before passing it on. Ultimately, secure network coding is a challenge of data integrity testing and verification at its most fundamental level. In many cases, standard data integrity approaches, such as cryptographic hash functions [13], message authentication codes [14], and digital signatures [15, 16], are used to ensure data integrity. It is necessary to verify every network codeword, which implies that it must be reviewed to guarantee that it has not been illegally modified. The problem is that routers combine packets in a linear method, and the resulting packets must be examined in the same way that the original packets were checked before they are combined. It is vital for the cryptographic approaches that are utilised to underpin contemporary network coding solutions to have a homomorphic property.

Our work.

Using this investigation as a starting point, researchers were able to uncover for the first time the relationship between safe cloud storage and secure network coding. If we use any publicly verifiable secure linear network coding method that is also publicly verifiable, we can create a publicly verifiable secure cloud storage mechanism. This connection allows many earlier solutions for secure network coding to be enhanced to enable secure cloud storage storage as a result of this connection. Our generic structure enables us to construct a variety of secure cloud storage systems that are compliant with existing secure network coding protocols by following a set of rules that we have outlined. Cloud data security is now being addressed on an as-needed basis, with procedures being developed as and when they are required. It is feasible to demonstrate the capabilities of our facility by utilising two upgraded cloud storage protocols that have been created in conjunction with our facility. In addition to providing new insights into public-key secure cloud storage protocols, these novel protocols, which are the result of our generic design, also provide fresh insights into private-key secure cloud storage techniques that were previously unavailable. For the first time, a secure cloud storage protocol that is secure in accordance with the first publicly verifiable secure cloud storage protocol is designed without the need to model a random hash function in the standard model, which is as described in the standard model, as described in the standard model. On top of all of that, we enhance our generic architecture to include more advanced features such as user anonymity and third-party public audits. There has been a significant lot of attention placed on these characteristics in recent years. Through the use of a security specification that simulates real-world application scenarios, we have demonstrated the security of our generic architecture.... Aside from that, we are currently creating and testing a functional prototype of a new, openly verifiable, and secure cloud storage technology that will be made available in the near future. The creation of a prototype paves the way for the implementation of the protocol. Our ultimate goal is to disseminate knowledge about cloud storage security solutions to the network coding community, and we feel that our work will contribute significantly to this effort.

2. BACKGROUND WORK

Secure Cloud Storage

In the age of cloud computing, clients may seek to use a cloud storage server to unload their massive amounts of data, which is a realistic request in this day and age. Cloud service providers may delete old data in order to save space, therefore clients must be convinced that their outsourced data is safe on a cloud server before enlisting their services. Cloud service providers may also delete old data in order to save space. Using a basic way of assuring data integrity, a client may be instructed to download the entire contents of the server and then validate each segment one at a time once the entire contents of the server has been downloaded. The inefficiency of this method, on the other hand, can be measured in terms of bandwidth use.

Building Blocks: PDP and POR.

As a result of this challenge, researchers have developed proofs of storage for various types of data. [3] When Ateniese and colleagues [3] proposed the concept of proved data possession (PDP), the client computed an authenticator (for example, a MAC) for each segment of her data (or file), and then uploaded the file, along with the authenticators, to the cloud service, to demonstrate ownership of her data. As part of an audit procedure, the client sends random segment-indices (challenge) to the server, which is then confirmed by the server. A common representation of the cardinality of a challenge is the function O(n), which we denote by the letter l in our notation. If the challenge requires it, the server performs specified computations on the stored data and then delivers a proof to the client, who can then use the proof to verify the integrity of her data. It includes the concept of public verifiability1, which enables data owners to delegate the task of auditing to a third-party auditor, hence reducing costs (TPA). As a result of having access to the public key, the TPA is in charge of conducting the audit. In private verifiable systems, the server's proof can only be confirmed by the client who has access to the secret key, which means that the server's evidence is not public. I'll give you an example of what I'm talking about in Figure 1.

For the first time, a formalised proof of retrievability (POR) for non-dynamic data has been established in a study by Juels and Kaliski [23]. (a similar idea is given for sublinear authenticators by Naor and Rothblum [30]). Specifically, according to Shacham and Waters [36], the primary assumption of POR is to encrypt the original file, authenticate the segments of encoded files, and then upload them to the storage server in the sequence indicated. It is necessary for the server to delete or alter a large number of data segments at the same time in order to delete or modify a single data segment. In this way, it is guaranteed that all of the file's segments may be recovered from the answers of a server that passes an audit but has a non-zero chance of failing the audit. Immediately following the release of Juels and Kaliski's work, an avalanche of POR techniques for both static and dynamic data began to emerge. A thorough list of POR systems can be found in [35], which is available online.

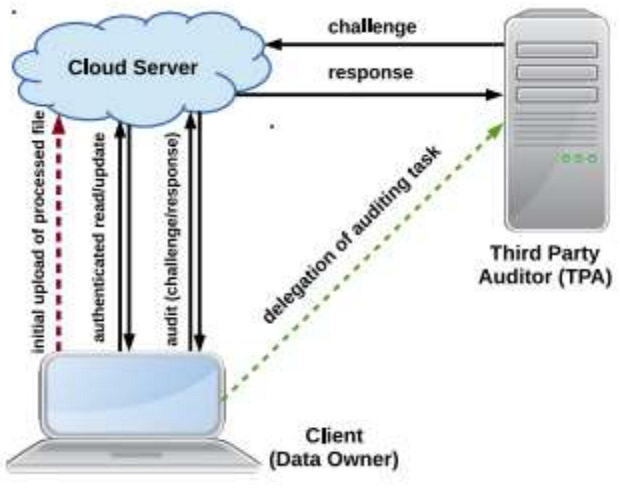
Cloud Server
challenge
response
initial upload of processed file
authenticated read/update
audit (challenge/response)
delegation of auditing task
Third Party
Auditor (TPA)
Client
(Data Owner)

Figure 1 depicts the architecture of the protocol for secure cloud storage, which is comprised of a set of rules. Once processing is complete, the client transfers the file to the cloud server, which stores it there for safekeeping. Furthermore, she has the power to make changes to her own records in addition to reading data from external sources. If the scheme is private verifiable, it is possible for the client to audit the data if he or she has access to the secret key used to encrypt the information (through challenge and response). Audits can be carried out by a third-party administrator (TPA) on behalf of the customer in line with a process that is publicly verifiable.

As a result, because we will only be working with a single cloud server, we will only explore a few safe cloud storage choices in a distributed setting in this study. For example, Curtmola et al. [15] (who utilised replication of data) and Bowers et al. [8] (who used a randomised controlled trial) have both employed replication of data (using error-correcting codes and erasure codes). Alternatively, cloud storage systems, which are now widely available, have found application across a wide range of industries [28, 34].

Security of an SSCS Protocol

In order to perform successfully, an SSCS protocol must possess the characteristics listed below [36].

1. Authenticity

Unless the cloud server is presented with an invalid proof of storage T (corresponding to the challenge set Q), it will be impossible for it to fabricate one without storing the challenged segments and their respective authentication information in their original form, save for the chance of occurrence insignificant in, which is required for storage authenticity.

2. Extractability

There is an extractor algorithm E that can extract (at least) the challenged segments (except when faced with negligible probability) by challenging A for a polynomial-time number of times and verifying that the responses sent by A can be constructed if the adversary A can correctly respond to a challenge Q with some nonnegligible probability. In other words, E has non-black box access to A because A is made available to him. Therefore, E has the ability to go back in time and rewind A whenever it is required.

The use of PDP in conjunction with SSC protocols ensures that practically all segments of file F may be retrieved without difficulty. POR-based SSCS processes, on the other hand, secure the extraction of all of F's segments by utilising erasure codes to ensure that no segments are missing during the extraction process.

General Construction of an SSCS Protocol from an SNC Protocol

A generic safe cloud storage strategy for static data has been developed by Chen et al. [14], which makes use of a secure network coding mechanism to protect the data stored in a cloud. n blocks are contained within each of the m vectors (or packets) that comprise the data file F that has been saved on the server. Each vector in the data file F has n blocks, which means that the data file F contains n blocks in total (or packet). This is the fundamental concept that must be adhered to in order to save these vectors on the server without augmenting them with unit vectors. During a client-server communication session, the client sends the server an element-by-element subset of the index set. Auditing is carried out in this manner as part of the overall auditing process (1, 2,...,m). These vectors are delivered to the client along with their unique tagging code after they have been augmented with the corresponding unit vectors and merged linearly in an authorised way. In order to ensure that the tag received is genuine, at the end of the process, the client must compare it to the vector that was provided with it. So the server serves as a mediator, while the client serves as both a transmitter and receiver at the same time, as seen in Figure 1. (or the next intermediate router).

Construction Of An SCS Protocol For Dynamic Data Using An SNC Protocol

Previously, we explored how Chen et al. [14] modified an SNC protocol to produce an SSCS protocol. This page continues the discussion. It is possible to set the maximum number of packets (or vectors) in a file that must be transferred across a network when employing a secure network coding approach to a specified quantity. This is due to the fact that it is important to know in advance how long the coefficient vectors required to augment the original vectors will be, which makes the operation far more difficult. This form of arrangement can be advantageous for any type of static data in general, and it is not limited to just spreadsheets. Amazon Web Services offers a secure cloud storage protocol for dynamic data, which allows customers to make changes to their data after it has been uploaded to a cloud server. This protocol is provided by Amazon Web Services. Are there any prospects for creating an SNC-enabled dynamic cloud storage protocol (DSCS) that is both efficient and safe? DSCS stands for dynamic cloud storage protocol. In this section, we'll discuss the issues listed above in more detail.

3. SYSTEM ANALYSIS

Modeling Secure Cloud Storage

Our cloud storage system is depicted in Figure as a series of representations. In the future, a user and a cloud will coexist peacefully. Individuals and organisations can utilise a computer, a mobile phone, or any other device; cloud service providers such as Amazon S3, Dropbox, Google Drive, and other similar services can also use any of these devices, as can individuals and businesses. To get started, a user uploads their files to a cloud storage service such as Dropbox. Afterwards, the user conducts regular audits to ensure that the data that is being outsourced is accurate and up to date, and the process is repeated. Regardless of whether the user determines that the proof provided by the cloud is authentic or invalid, the data will not be compromised; nonetheless, extra action (which is outside of our control) may be required, such as a lawsuit or data recovery. If a user has the ability to examine the integrity of the data saved in a cloud storage system, the data stored in the cloud storage system should be accurate, safe, and efficient.

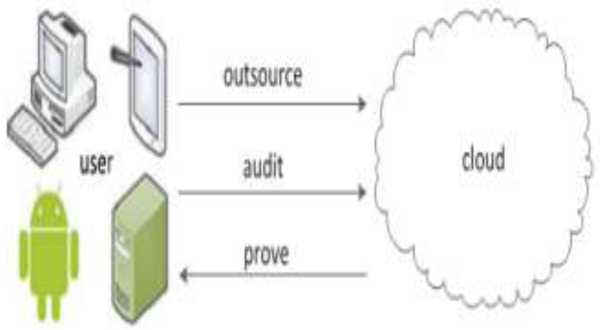

Fig. A Secure Cloud Storage System.

A High-Level Protocol Specification for Secure Cloud Storage

To determine whether or not a cloud is real, the user will be required to have a tiny quantity of secret data on its side that has been computed according to a set level of protection in order to do an audit query. A number represents the level of security, and the letter K represents the secret material, while the letter F represents information about the subject. The function F is processed with the help of the variable k. F′ is a representation of data that has been processed and is then transferred to the cloud, where it is authenticated using the data that has been processed. When a user submits an audit query to the cloud, the cloud uses the stored information F′ to construct an evidence G that indicates that the data is not corrupted or otherwise compromised. The consumer then tests G to determine whether or not it is authentic. When the verification result for a specific person has been obtained, the d symbol should be used to signify this. A number of extremely sophisticated algorithms are employed to safeguard the information stored on our cloud storage system. There are algorithms such as Key Generation and the Outsource Algorithm, among others, that may be found in this section.

Key Generation(λ) → k: The operation is carried out by the user with the assistance of a security parameter in order to generate a secret key K for auditing and verification reasons.

Outsource(f, k) → F′: In order to obtain the processed data F′, it is necessary for the user to run an algorithm on the input data F while using a secret key K as the input key. Data F's authentication information is included in the processed data, which is then uploaded to the cloud, and can be found there.

Audit(k) → q: The user will be required to perform this algorithm in order for an audit query q to be prepared and sent to the cloud on their behalf.

Prove(q, F′) → Γ: The a• cloud is prompted by an auditing inquiry q to use the previously stored data F′ to provide a proof in response to the query.

Verify(q, Γ;K) → δ: A user can use the secret key K to verify that the cloud's evidence is valid when an audit query q and the cloud's proof are both entered. The secret key K can be used to verify that a query and the cloud's proof are both valid. If this is not the case, then the value = 0 is returned instead. It is expected that the user will receive a response with a value of 1 if the proof is authentic.

Secure Network Coding Model

Our secure network coding model is made up of three parts: the sender, the router, and the recipient, which we will discuss in more detail later. As illustrated in Figure 1, senders attempt to broadcast data to a large number of recipients at the same time. Packetized data is sent through a network from one site to another as a linear combination of packets from the transmitter to the receiver.. An additional feature of the network router is that it sends a linear combination of the data packets it has received to its next hop in the network. Once encoded data packets have been received, it is the responsibility of the receivers to decode them in order to recover the original data that was contained inside them. The security authentication information associated with each and every packet of data is included in order to prevent the data from being altered during transmission by a hostile router. Once the integrity of received packets has been validated, the router proceeds to combine the required data packets before sending the combined packet of data and secure authentication information to the intended destination. According to the regulations of a given protocol, the method by which the combined authentication information is computed and generated is determined.

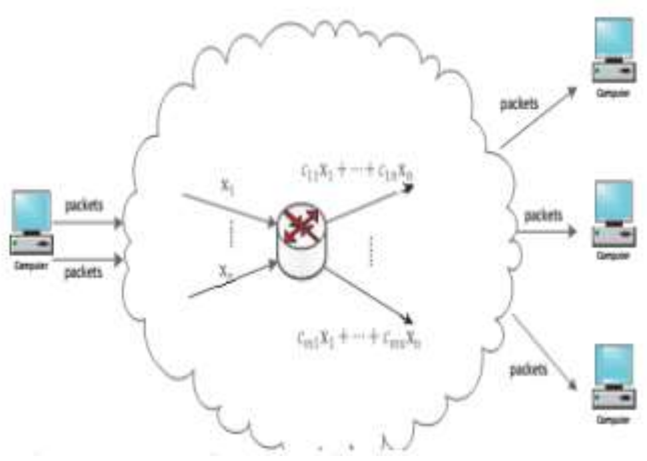

Fig. Typical use of Network Coding.

A High-Level Protocol Specification for Secure Network Coding

Specifically, we deploy four incredibly efficient secure network coding methods for the purpose of carrying out our network coding protocol implementation. Those are the algorithms that are used for a variety of tasks, including establishing keys, authenticating users, combining data, and verifying the integrity of the data, among others.

Key Generation λ → SK, : Following receipt of a security parameter from the receiver, a secret key (SK) and a public key (PK) are produced by the sender in order to enable packet authentication following receipt of the parameter.

Authentication xi• ; SK → x ,i In order to communicate with the receiver, the sender computes an authentication ti and transmits a packet to the network called xi Fp n+m based on the information obtained from the receiver (xi ,ti).

Combine ui• ,ti i=1,…,l , c1, … , cl → w.t : When the authentication information for a group of packetsui+FP n+m is received, the group of packetsui+FP n+m is divided into two groups, one for each group of packetsui+FP n+m that has been authenticated.

Verify(w,t) → δ: Specifically, the router in question employs this approach in order to generate the combined packet with the coefficients c1,...,cl and the combined authentication information t, as well as the combined authentication information t, as well as the combined authentication information t.

When a packet is received, the value of w FP• n+m w is changed. A packet must be examined using an algorithm in order to detect whether or not it has been maliciously altered by a receiver or a router. If the packet is accurate, a value of one will be returned; if the packet is wrong, a value of zero will be returned.

4. CONCLUSION

It is the first time that we have established a relationship between secure cloud storage and secure network coding in this paper. A methodical strategy for building an open-source secure cloud storage protocol that may be implemented using any secure network-coding technology is proposed in this context. The development of a secure cloud storage approach that does not rely on the random oracle heuristic for data storage has been completed. On top of that, we are strengthening our generic design in order to allow users to remain anonymous and for other parties to undertake public audits of our efforts. With confidence, we believe that our open-source prototype will pave the way for the widespread deployment of secure cloud storage protocols in the real world in the future. On the basis of our general construction, as well as present and future research on secure network coding, which will be built on our general construction, it is possible to design new and efficient secure cloud storage protocols in the future. Additionally, this can be done in the opposite direction to test whether or not a secure cloud storage protocol can be constructed from an existing secure network coding technique under specific circumstances. If this is the case, it is feasible that more characteristics will be added to the latter product.

REFERENCES:

❖ 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.

- 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.
- Juels and B. S. Kaliski, "PORs: Proofs of retrievability for large files," in ACM Conference on Computer and Communications Security, 2007, pp. 584–597.
- H. Shacham and B. Waters, "Compact proofs of retrievability," Journal of Cryptology, vol. 26, no. 3, pp. 442–483, 2013.
- Juels and B. Kaliski Jr, "PORs: Proofs of retrievability for large files," in Proc. ACM Conf. Comput. Commun. Security, 2007, pp. 584–597.
- Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacypreserving public auditing for secure cloud storage," IEEE Trans. Comput., vol. 62, no. 2, pp. 362–375, Feb. 2013.
- N. Cai and R. W. Yeung, "Secure network coding," in Proc. IEEE Int. Symp. Inf. Theory, 2002, p. 323.
- 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.
- R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. IEEE Transactions on Information Theory, 46(4):1204–1216, 2000.
- S.-Y. Li, R. W. Yeung, and N. Cai, "Linear network coding," IEEE Trans. Inf. Theory, vol. 49, no. 2, pp. 371–381, Feb. 2003.
- Q. Li, J. C. Lui, and D.-M. Chiu, "On the security and efficiency of content distribution via network coding," IEEE Trans. Dependable Secure Comput., vol. 9, no. 2, pp. 211– 221, Mar./Apr. 2012.