Secure Outsourced Data Transfer with Integrity Verification in Cloud Storage

Jianbing Ni\textsuperscript{1}, Xiaodong Lin\textsuperscript{1}, Kuan Zhang\textsuperscript{1}, Yong Yu\textsuperscript{3} and Xueming (Sherman) Shen\textsuperscript{1}
\textsuperscript{1}Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1, Email: \{j25ni, k52zhang, sshen\}@uwaterloo.ca
\textsuperscript{2}Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, Ontario, Canada L1H 7K4, Email: xiaodong.lin@uoit.ca
\textsuperscript{3}School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China 611731, Email: yuyong@uestc.edu.cn.

Abstract—With the proliferation of cloud storage, outsourced data transfer becomes an essential requirement for users to migrate their outsourced data from one cloud to another. However, data confidentiality and integrity are big concerns for users when their data are migrating between two semi-honest clouds. In this paper, we propose a secure outsourced data transfer scheme (SODT) to achieve secure data migration in cloud storage. SODT allows users to migrate the remote data from one cloud to another without retrieving the data from the former cloud, such that the data confidentiality and integrity can be achieved during this process. In addition, the cloud can perform secure data erasure after the data are migrated by utilizing the proxy re-encryption technique. Finally, we discuss the security properties including confidentiality and integrity of SODT, and demonstrate its efficiency in terms of the computational and communication overhead.

I. INTRODUCTION

Cloud storage \[1\] allows users to remotely store their data on cloud servers and offers the on-demand high quality applications and services from a shared pool of configurable resources, without the burden of local data storage and maintenance. An increasing number of individuals and corporations outsource their data to cloud servers for saving the cost of purchasing and maintaining storage devices, and enjoying the flexible data access at anytime and from anywhere. 82\% of organizations embracing cloud storage services benefit for reducing cost of IT resources \[2\]. The number of tenants in cloud storage is predicted to reach 2 billion by 2018 \[3\].

Due to the promising market prospect, a large number of cloud storage services have emerged, e.g., SkyDrive, Dropbox, ZipCloud and OneDrive, providing various levels of services on access rate, security, reliability and prices \[4\]. This competitive phenomenon not only provides users with multiple options when they plan to outsource data, but also brings opportunities to shift the cloud storage services if they do not satisfy their current storage services. In addition, cloud data are frequently transferred across multiple data centers to guarantee the multiple replication storage and sharing. According to Cisco report \[3\], the traffic among different clouds is predicted to reach 9\% of total cloud traffic by 2018. One straightforward solution to achieve the data transfer for users is to retrieve their outsourced data from one cloud and upload them to the new one. Unfortunately, the users do not have enough storage resources to temporarily store their retrieved data. Therefore, outsourced data transfer becomes a fundamental requirement for users to migrate their data from one cloud to another.

However, it is of significant difficulty to achieve secure data transfer between two clouds, as both cloud storage providers may behave unfaithfully over the data \[5\], \[6\]. First of all, the integrity of the outsourced data may be invaded by the cloud storage providers \[7\], \[8\]. For example, they may discard the data that are rarely or never visited to rent the space to other users for economic reasons. Furthermore, a lazy cloud may migrate only a portion of user’s required data, since data transfer consumes many resources and bandwidth. Secondly, it is difficult to prevent the data exposure to the cloud storage providers \[9\]. Once the data are uploaded to the cloud, the users lose the physical control over their data. All the data are exposed to the cloud storage providers and the providers can arbitrarily access users’ data. Thirdly, the cloud can keep the copies of transferred data to discover potential value, leading to the data disclosure accidents. In addition, the infrastructure of cloud storage are confronted with the broad range of threats from external attackers towards data integrity and confidentiality \[10\]. For instance, attackers may maliciously access the users’ outsourced data by means of system vulnerabilities and software bugs, or eavesdrop on the communication channels to capture the users’ data, when they are transferring between two clouds. In summary, the success of cloud architecture is impeded, if there is no security guarantee on the outsourced data during data storage and transfer processes.

To achieve data confidentiality and integrity, in this paper, we propose a novel secure outsourced data transfer scheme (SODT) from polynomial-based authenticators \[11\] and the BCP encryption scheme \[12\]. Specifically, our contributions are summarized as follows:

- We propose SODT to achieve secure data transfer and prevent outsourced data disclosure in cloud storage. In SODT, we improve the BCP encryption scheme \[12\] to support proxy re-encryption and use it to realize the confidentiality of the outsourced data. Utilizing the homomorphism of improved BCP encryption scheme

...
and polynomial-based authenticators [11], the users can efficiently verify the integrity of their data in clouds to ensure the secure outsourced data transfer. In addition, the users can discard the transferred data in cloud by means of the proxy re-encryption technique [13].

- Through the security analysis, we prove that SODT achieves the properties of data confidentiality, data integrity verification, secure data transfer and secure data erasure. We also demonstrate the efficiency of SODT in terms of the computational and communication overhead.

The remainder of this paper is organized as follows: In section II we show the related work. We state the problem of outsourced data transfer in section III. In section IV we propose our SODT scheme, which follows security analysis in section V and performance evaluation in section VI. Finally, we draw the conclusion in section VII.

II. RELATED WORKS

To achieve the data integrity in cloud storage, several provable data possession schemes [6], [14] have been proposed, where users are allowed to verify the integrity of the cloud data without retrieving the data from the cloud. To realize secure data deletion, Reardon et al. [15] introduced a general data deletion approach by utilizing encryption and key wrapping techniques and defined a key disclosure graph to model the key generation wrapping. Feng et al. [16] proposed a secure data erasure scheme with public verifiability employing a Trusted Platform Module. Unfortunately, these schemes only guarantee that the data are deleted irrecoverably from a physical medium.

To prevent the data disclosure during migration, Cloudsfer [17] utilizes data encryption technique to build a secret channel between two clouds. Although Cloudsfer provides data protection during the data transfer, it cannot ensure the data integrity and correctness for the data in clouds. To address these issues, Yu et al. [18] proposed a provable data possession scheme supporting secure data transfer for cloud storage. They utilize the privately verifiable provable data possession scheme [19] to allow users to check the integrity of their data in clouds and design a randomization technique to prevent the data disclosure and achieve secure data erasure. In Yu et al. scheme, the binary length of the values, which are used to randomize the transferred data, is equal to the size of the data. In this paper, we propose a more efficient scheme from the BCP encryption scheme, polynomial-based authenticators and proxy re-encryption technique to realize the cloud data integrity, secure data transfer and erasure.

III. PROBLEM STATEMENT

In this section, we present the system model, security threats and identity security goals.

A. System Model

The system model of outsourced data transfer involves two entities: users and clouds. The users have huge volume of data to store, but limited data storage resources and computing capability. They prefer to rent the virtual storage spaces offered by cloud storage providers, rather than purchase physical storage facilities. The clouds have a large amount of storage spaces and computing resources, and offer remote data storage services to users in a pay-as-you-use manner. As illustrated in Fig. 1, when a user wants to outsource the data, he/she chooses a cloud (e.g., Cloud A), rents the storage space and uploads the data to Cloud A. After the user enjoys the storage service for a period of time, he/she may not satisfy it or find a better cloud storage service. As a result, the user rents the storage space provided by a new cloud storage provider (e.g., Cloud B), and migrates the outsourced data from Cloud A to Cloud B. To ensure the data integrity during these processes, the user utilizes a remote data integrity checking protocol to verify the integrity of the cloud data.

B. Security Threats

Security threats towards users’ outsourced data are from both internal and external attacks. The external attackers, such as motivated hackers, may corrupt the outsourced data when they are maintaining on the cloud servers or transmitting on channels. The internal attackers, including the honest-but-curious cloud storage providers, may delete the data that are rarely or never accessed on the servers. Specifically, the following security threats should be taken into account.

- **Data privacy exposure:** Data confidentiality is one of the most critical security threats for users in cloud storage in the following reasons. Firstly, since the cloud servers are maintaining all data, the employees can illegally acquire the managers privileges to access the data via system vulnerabilities. Secondly, the clouds may share the maintaining data to their corporators or outsource the data to subcontractors. Last but not least, since the data from multiple users are maintaining on the same physical server, some users may maliciously read the data of other users across the virtual machines. Therefore, the outsourced data are vulnerable to various threats in clouds, and all the data are exposed to the public once they are uploaded to the cloud servers.
• **Data corruption:** Due to the frequent data access and update operations, the probability of data errors occur increases and the life time of storage medium becomes short. Thus, the data loss accidents may happen unexpectedly and the cloud storage providers try to hide these accidents to maintain a good reputation. The clouds may also delete some data that are rarely or never visited to rent the spaces to other users for monetary benefits. In addition, Cloud A may send a portion of the required data to Cloud B and claim that all the data are transferred, or send forged data to cheat the user.

• **Copy reservation:** The cloud may reserve the transferred data and try to discover some implicit benefits from them. This behavior is unexpected from users’ points of view. Besides, the copy reservation may cause the disclosure of the sensitive information about users.

Due to the competitive relationship, we assume both clouds would not collude to cheat users. Otherwise, they can be viewed as one cloud and the outsourced data transfer is meaningless. There are regulations and policies to guide the clouds to follow the protocols agreed with users. Specifically, the clouds honestly store the data, perform the data integrity checking protocol and transfer the data as required by the users. As the data are business asset for the clouds, there are financial incentives to build firewalls or intrusion detection mechanisms for protecting the stored data from being accessed by the external attackers. Furthermore, Cloud B is rational during the data transfer process, indicating that it tries to accomplish the data transfer successfully, rather than actively corrupting the transferring data to slander Cloud A. This assumption is reasonable since the goal of Cloud B is to provide data storage service for the users, and it cannot obtain any benefit from slandering Cloud A, as the user has already stopped the data storage service of Cloud A. In addition, to prevent attackers from breaching data, all data are transmitted on the authenticated channels (e.g. SSL/TLS channels [20]).

### C. Security Goals

To achieve secure outsourced data transfer under the aforementioned model, our scheme should achieve the following security goals.

• **Data confidentiality:** To prevent both the internal and external attackers from obtaining users’ data, all the data stored on cloud servers should be encrypted before uploading to the cloud.

• **Data integrity verification:** To ensure the clouds are possessing the data correctly, there should be a remote data integrity verification protocol for the users. The cloud cannot convince the users that all data are well maintained on the servers, if some data have been corrupted already, unless the cloud can guess all the missing data.

• **Secure data transfer:** To guarantee all the transferring data can reach Cloud B successfully, Cloud B and the users should be able to verify the data correctness and integrity. If some data are lost during the transmission or Cloud A sends a portion of the data, Cloud B can refuse to receive the data and inform the accident to the users.

• **Secure data erasure:** There should be an effective mechanism to assure that the copies in Cloud A are discarded after the data are transferred to Cloud B. Thus, the users have no worries about the data leakage due to the copy reservation in Cloud A.

### IV. PROPOSED SODT SCHEME

To achieve the outsourced data confidentiality and integrity verification, the users encrypt their data blocks before uploading them to Cloud A by utilizing the improved BCP encryption scheme and compute the polynomial-based tags for the data blocks. During the remote data integrity checking, the clouds can aggregate the ciphertexts of data blocks and the corresponding tags to generate a proof with a constant size to show the correctness and integrity of the data. During the data transfer, the rational Cloud B can verify the data integrity by means of batch verification and accept the uncorrupted data. In addition, to guarantee secure data erasure on Cloud A, we employ the proxy re-encryption technique [13] to re-encrypt the data on behalf of Cloud B and discard the old decryption key. Thus, the data in Cloud A are unencryptable and no attacker is able to recover any valuable information from these copies.

Let $N = pq$ be an RSA modulus and $\phi(N) = (p-1)(q-1)$, where $p, q$ are primes of the form $q = 2q' + 1, p = 2p' + 1, p', q'$ are also primes. $G = QR_{N^2}$ is a cyclic group of quadratic residues modulo $N^2$ and $\text{ord}(G) = \lambda(N^2)/2 = pp'q'q = N\lambda(N)/2$, where $\lambda(N) = 2p'q'$. $H : \{0, 1\}^* \rightarrow \mathbb{Z}_N$ is a cryptographic hash function, $\delta : \{0, 1\}^* \rightarrow \mathbb{Z}_N$ is a pseudo-random functions (PRF) and $\pi : \{0, \ldots, n - 1\} \times \mathbb{Z}_N \rightarrow \{0, \ldots, n - 1\}$ is a pseudo-random permutation (PRP).

**KeyGen.** A user $U$ randomly picks $\gamma \in \mathbb{Z}_{N^2}$ and $a \in [1, \text{ord}(G)]$ to generate $g = \gamma^2 \mod N^2$ and $h = g^a \mod N^2$. $U$ also chooses a random large prime $e$ as the public key, such that $e > N$, $gcd(e, \phi(N)) = 1$ and generates the corresponding secret key $d = e^{-1} \mod \phi(N)$. In addition, $U$ chooses a random $\alpha \in \mathbb{Z}_N$ and generates a random signing key pair $(ssk, spk)$. Finally, the public key of $U$ is $Ppk = (g, h, e, spk)$ and the secret key is $Sk = (a, d, \alpha, ssk)$.

**Store.** When $U$ wants to upload a file $M'$ to Cloud A, $U$ firstly uses a coding method, such as Reed-Solomon code, to generate the file $M$. Then, $U$ divides $M$ into $n$ blocks and each block is further split into $s$ sectors: $M = \{m_{ij}\}_{0 \leq i \leq n - 1, 0 \leq j \leq s - 1} \in \mathbb{Z}_N$. For every sector $m_{ij}$, $U$ randomly chooses $r_{ij} \in \mathbb{Z}_N$ to compute the ciphertext $(A_{ij}, B_{ij})$ as

\[ A_{ij} = h^{r_{ij}} \mod N^2, \]
\[ B_{ij} = g^{r_{ij}}(1 + m_{ij}\alpha^j N) \mod N^2. \]

After that, $U$ picks a random $u \in \mathbb{Z}_N$ and a random file name $fname \in \mathbb{Z}_N$. $U$ generates a block tag for each data block $m_i$, $0 \leq i \leq n - 1$ as

\[ \sigma_i = (H(fname||u||i) \prod_{j=0}^{s-1} u^{m_{ij}\alpha^j})^d \mod N, \]
Then, the cloud computes and $A$ to Cloud B, on its servers. Cloud A deletes its data copy for the public key (e.g., Cloud B). $T_{\text{req}}$ (the new public key, and deletes the original key pair $\{a, h\}$.

Then, $U$ computes the proxy re-encryption key $rk = b/a$, and generates $T_{\text{req}} = T_{\text{req0}}||rk||Sig_{ssk}(T_{\text{req0}}||rk)$. Finally, $U$ sends $T_{\text{req}}$ to Cloud A.

When receiving $T_{\text{req}}$ from Cloud A, Cloud A firstly checks the validity of the signature $Sig_{ssk}(T_{\text{req0}}||rk)$ using the public key $spk$. If it is invalid, Cloud A returns failure and aborts; otherwise, it sends the corresponding file $\{fname, u, \{A_{ij}, B_{ij}\}_{0 \leq i < n-1, 0 \leq j \leq s-1}, \{\sigma_i\}_{0 \leq i \leq s-1}\}$ to Cloud B, along with the transfer request $T_{\text{req}}$.

Upon receiving the data from Cloud A, Cloud B firstly verifies the signature $Sig_{ssk}(T_{\text{req0}}||rk)$. If it is invalid, Cloud B returns failure and aborts; otherwise, it randomly picks $n$ elements $(w_0, \ldots, w_{n-1}) \in \mathbb{Z}_N$, and computes

$$A = \prod_{j=0}^{s-1} \prod_{i=0}^{n-1} (A_{ij})^{w_i} \mod N^2,$$

$$B = \prod_{j=0}^{s-1} \prod_{i=0}^{n-1} (B_{ij})^{w_i} \mod N^2,$$

$$\sigma = \prod_{i=0}^{n-1} \sigma_i^{w_i} \mod N.$$

Then, Cloud B returns $A$ to $U$. $U$ calculates $\widehat{A} = A^{-a} \mod N^2$ and replies $A$ to Cloud B. Upon receiving $A$, Cloud B computes $\mu = (B/\widehat{A} - 1 \mod N^2)/N$ and checks whether the following equation holds:

$$\sigma^e = \prod_{i=0}^{n-1} H(fname||u||i)^{\nu_i} u^\mu \mod N. \quad (1)$$

If not, Cloud B requires Cloud A to re-transmit the data; otherwise, it informs the user $U$ and Cloud A that the data transfer is successful, and utilizes the proxy re-encryption key $rk$ to compute $A_{ij} = (A_{ij})^{rk} \mod N^2$ for $0 \leq i \leq n-1, 0 \leq j \leq s-1$. Finally, Cloud B stores $\{fname, n, u, \{A_{ij}, B_{ij}\}_{0 \leq i \leq n-1, 0 \leq j \leq s-1}, \{\sigma_i\}_{0 \leq i \leq s-1}\}$ on its servers. Cloud A deletes its data copy on Cloud B.

Integrity Check. To check the data possession of the file $M'$ on Cloud A or B, $U$ picks a value $c$ and randomly chooses $k_1, k_2 \in \mathbb{Z}_N$, which denotes the number of the challenged data blocks. Finally, $U$ sends the challenge $\text{chal} = (c, k_1, k_2)$ to the cloud (Cloud A/B).

Upon receiving $\text{chal}$ from $U$, the cloud generates $i = \pi_{k_1}(t)$ and $\nu_i = k_{2i}(t)$ for $0 \leq t \leq c - 1$. Let $Q$ be the set $\{(i, \nu_i)\}$. Then, the cloud computes

$$A = \prod_{j=0}^{s-1} \prod_{\{(i, \nu_i)\} \in Q} (A_{ij})^{\nu_i} \mod N^2,$$

$$B = \prod_{j=0}^{s-1} \prod_{\{(i, \nu_i)\} \in Q} (B_{ij})^{\nu_i} \mod N^2,$$

$$\sigma = \prod_{\{(i, \nu_i)\} \in Q} \sigma_i^{\nu_i} \mod N.$$

Finally, the cloud returns the proof $P = (A, B, \sigma)$ to $U$.

Upon receiving the proof $P$ from the cloud, the user $U$ uses the secret key $x$ ($x = a$ if Cloud A is queried, $x = b$ otherwise) to decrypt $(A, B)$ by computing $\mu = (B/(A^{-x}) - 1 \mod N^2)/N$. $U$ checks whether the following equation holds. If yes, the data on the cloud are intact; otherwise, the data have been corrupted, such that the user can investigate and affix legal liability of the corresponding cloud.

$$\sigma^e = \prod_{\{(i, \nu_i)\} \in Q} H(fname||u||i)^{\nu_i} u^\mu \mod N. \quad (2)$$

The correctness of data encryption is elaborated as follows.

$$\begin{align*}
\mu &= ((B/(A^{-b}) - 1 \mod N^2))/N \\
&= \left(\prod_{j=0}^{s-1} \prod_{\{(i, \nu_i)\} \in Q} (B_{ij}/(A_{ij})^{-b})^{\nu_i} - 1 \mod N^2\right)/N \\
&= \left(\prod_{j=0}^{s-1} \prod_{\{(i, \nu_i)\} \in Q} (B_{ij}/A_{ij}^{-b})^{\nu_i} - 1 \mod N^2\right)/N \\
&= \left(\prod_{j=0}^{s-1} \prod_{\{(i, \nu_i)\} \in Q} (1 + N)^{\nu_i m_i^2 a^j} - 1 \mod N^2\right)/N \\
&= \sum_{j=0}^{s-1} \sum_{\{(i, \nu_i)\} \in Q} \nu_i m_i^2 a^j.
\end{align*}$$

If $\mu$ is obtained, the equation (1) is satisfied since

$$\begin{align*}
\sigma^e &= \left(\prod_{\{(i, \nu_i)\} \in Q} \sigma_i^{\nu_i} \right)^e \mod N \\
&= \left(\prod_{\{(i, \nu_i)\} \in Q} H(fname||u||i)^{\nu_i} u^{\nu_i m_i^2 a^j} \right)^e \mod N \\
&= \prod_{\{(i, \nu_i)\} \in Q} H(fname||u||i)^{\nu_i} u^{\nu_i m_i^2 a^j} \mod N \\
&= \prod_{\{(i, \nu_i)\} \in Q} H(fname||u||i)^{\nu_i} u^{\nu_i m_i^2 a^j} \mod N.
\end{align*}$$

V. Security Analysis

We analyze security requirements described in Section III-C, namely, data confidentiality, data integrity verification, secure data transfer and secure data erasure.
Data confidentiality: To keep the data confidentiality, the user encrypt each sector \(m_{ij}\) to generate the ciphertext \((A_{ij}, B_{ij})\) using the method derived from the BCP encryption. Our modified BCP encryption is semantic security related with the Decisional Diffie-Hellman problem in \(\mathbb{Z}_{p^2}\) [12]. Decisional Diffie-Hellman problem in \(\mathbb{Z}_{p^2}\) is that given \(g, g^a, g^b, g^c \in \mathbb{Z}_{p^2}\), to determine if \(c = ab \pmod{\text{ord}(G)}\) or not. We say that a problem is hard if, for every probabilistic polynomial time algorithm, the probability to solve this problem is negligible.

**Theorem 1 (Semantic Security).** The modified BCP encryption is semantic security if and only if the Decisional Diffie-Hellman (DDH) problem in \(\mathbb{Z}_{p^2}\) is hard.

**Proof.** If there is a polynomial-time adversary \(A\) to break the security of the modified BCP encryption, then one can use this capacity to solve the DDH problem in \(\mathbb{Z}_{p^2}\) [12]. Our goal is to use \(A\) to decide whether \((g, g^a, g^b, g^c) \in \mathbb{Z}_{p^2}\) is a Diffie-Hellman tuple or a random one. The public key is set as \((N, g, h)\), where \(h = g^a\). \(A\) chooses the messages \(m_0\) and \(m_1\), and we flip a coin to get \(d \in \{0, 1\}\) and encrypt \(m_d\) to obtain \((A, B) = (g^d, g^{d(l+1+m_dN)} \mod N^2)\).

Obviously, if \(c = ab \pmod{\text{ord}(G)}\), then \((A, B)\) is a valid ciphertext of \(m_d\); otherwise, \((A, B)\) is a random ciphertext. Let \(b = ca^{-1}l\), where \(l\) is distributed random and uniformly distributed in \([1, \text{ord}(G)]\) and we can denote \(l = l_1 + l_2\lambda(N)/2\), with \(l_1, l_2 \in \mathbb{Z}_N\). So the ciphertext received by \(A\) is

\[
A = g^d \mod N^2, \quad B = g^{ca^{-1}l}(1 + m_d N) \mod N^2.
\]

We concentrate on the value of \(B\), that is,

\[
B = g^{ca^{-1}l}(1 + m_d N) \mod N^2
= g^{ca^{-1}l} g^{l_2\lambda(N)/2} (1 + m_d N) \mod N^2
= g^{ca^{-1}l} l_2 (1 + m_d N) \mod N^2
= g^{ca^{-1}l} (1 + (l_2 + m_d N)) \mod N^2.
\]

Note that, \(l_2\) hides \(m_d\) perfectly and thus \(A\) cannot guess \(d\) better than at random.

Data integrity verification: The data integrity verification ensures that it is negligible for a polynomial-time adversary \(A\) to generate a proof successfully without possessing all challenged blocks. We prove that if \(A\) can construct a valid proof without storing all the blocks corresponding to a given challenge, then we can solve the RSA problem [21], that is, given an RSA public key \((N, e)\) and a value \(y = x^e \mod N\), to compute \(x\), with the involvement of Lemma 1.

**Lemma 1 [19].** Given \(x, y \in \mathbb{Z}_N\) and \(a, b \in \mathbb{Z}\), such that \(x^a = y^b\) and \(\gcd(a, b) = 1\), one can efficiently compute \(\bar{x} \in \mathbb{Z}_N\) such that \(\bar{x}^a = y\).

**Theorem 2.** There is no polynomial-time adversary that can cause the user to accept an integrity verification instance with non-negligible probability, when the challenged data blocks have been corrupted, if the RSA problem [21] is hard.

**Proof.** Given the RSA public key \((N, e)\), along with a value \(y \in \mathbb{Z}_N\), our goal is to find \(x \in \mathbb{Z}_N\), such that \(x^a = y \mod N\). We prove that if there is an adversary \(A\) can generate a valid proof that succeeds the equation (1) without possessing all queried blocks, thus, we can output \(x \in \mathbb{Z}_N\) that satisfies \(x^a = y \mod N\). Suppose \((A', B', \sigma')\) is \(A\)'s response that is not equal to the expected response \((A, B, \sigma)\) generated from all correct blocks. Due to the security of the modified BCP encryption, \((A', B')\) is the ciphertext of \(\mu'\) and \((A, B)\) is the ciphertext of \(\mu\), respectively. Due to the correctness, the expected response satisfies the equation (1), that is,

\[
\sigma = \prod_{(i, \nu_i) \in Q} H(f_{\text{name}}(\|u||i)^{\nu_i} u^\mu \mod N).
\]

The adversary’s response is also assumed to satisfy the equation (1), that is,

\[
\sigma' = \prod_{(i, \nu_i) \in Q} H(f_{\text{name}}(\|u||i)^{\nu_i} u^{\mu'} \mod N).
\]

If \(\mu = \mu'\), then \(\sigma = \sigma'\), so \(\mu \neq \mu'\) and \(\sigma \neq \sigma'\). Thus, we divide the first equation by the second one, and have

\[
\frac{\sigma}{\sigma'} = u^{\mu' - \mu} \mod N.
\]

Set \(u = \tau e y^{\eta}\), where \(\tau \in \mathbb{Z}^*_N\) and \(\eta \in [1, \lambda(N)]\), and obtain

\[
(\frac{\sigma}{\sigma'}) \cdot (\tau^{\mu' - \mu})^e = y^{\eta(\mu' - \mu')} \mod N.
\]

Provided that \(\gcd(e, \eta(\mu' - \mu')) = 1\), we can recover the value \(x\), such that \(x^e = y\), using the Lemma 1.

It remains to show that the probability of \(\gcd(e, \eta(\mu' - \mu')) \neq 1\) is negligible. Since \(e\) is a large prime and larger than \(N\), \(\eta\) and \(\mu' - \mu\) are less than \(N\), so \(\gcd(e, \eta(\mu' - \mu')) = 1\).

**Secure data transfer.** To ensure the data integrity during the data transfer, Cloud B checks the validity of each tag using batch verification. Since the tags \(\sigma_i\) are computed using RSA signature [22], no adversary is able to convince Cloud B to accept the corrupted data sent by Cloud A, if the RSA signature is unforgeable. Therefore, if the RSA signature scheme is existentially unforgeable, the data transfer is secure.

**Secure data erasure:** To delete the copy on Cloud A, Cloud B re-encrypts the transferring data with the proxy re-encryption key \(rk\) given by the user \(U\). The data on Cloud A, which is encrypted using \(g^a\), is transformed to be the ciphertexts that can only be decrypted using the new secret key \(b\). Therefore, no adversary can recover the data on Cloud A, after the user updates the secret-public key pair, if the proxy-encryption is secure, whose security has been proved to reduce to the Computational Diffie-Hellman assumption [23]. If Computational Diffie-Hellman assumption [23] is hold, the data erasure is secure.

**VI. PERFORMANCE EVALUATION**

To analyze the computational overhead of SODT, we conducted an experiment on a notebook with Intel Core i5-4200U CPU @ 2.29GHz and 4.00GB memory. We use the MIRAcl library to implement number-theoretic based methods of cryptography. The RSA modulus \(N\) is approximately 1024 bits and the parameters \(p\) and \(q\) are both 512 bits. In Fig 2(a), we show the execution time of users to process a file with the size of 1GB, 2GB, 3GB, 4GB or 5GB in Store phase, and each file is divided in 262144, 65536, 16384, 4096, 1024 or 256 blocks, respectively. Fig 2(b) shows the running time of the cloud to receive the corresponding migrating file from another cloud in.
Transfer phase. If the number of blocks is larger than 16384, the execution time of users in Store and clouds in Transfer increases significantly. In Fig 3, we set $c = 300$ and $c = 460$, and demonstrate the time cost of the cloud when generating the proof to respond the integrity challenge, and the users that verify the validity of the proof in IntegCheck phase. If the number of sectors is less than 16384, the computations of the cloud to generate the proof are not time-consuming. The time cost of users to check the proof is mainly associated with the number of challenged blocks.

The communication overhead is relatively low for our scheme. Specifically, to achieve secure data transfer, the user sends a proxy re-encryption key $r_k$ to Cloud B to re-encrypt the transferring data. Thus, the communication overhead between the users and clouds is constant and is much lower than that in [18], in which the user needs to sends random values to update the transferring data, whose length is equal to that of the transferring data. In the IntegCheck phase, the length of the proof $P = (A, B, \sigma)$ is also constant, unlike the schemes in [9, 10], the length of proofs linearly varies with the number of sectors $s$.

VII. CONCLUSION

In this paper, we have proposed a secure outsourced data transfer scheme to protect the users’ outsourced data when they are transferring from one cloud to a new one. In the proposed scheme, we have developed an enhanced BCP encryption to encrypt the users’ data to prevent the sensitive information from being disclosed, and the polynomial-based authenticators to achieve efficient remote data integrity verification. The proxy-re-encryption technique is employed to securely and sufficiently discard the transferred data in cloud. We have also analyzed the security properties and proved that the security of our proposed scheme depends on the mathematically hard problems. Finally, we have demonstrated the efficiency of our scheme by conducting an experiment. For the future work, we will design a secure data transfer scheme with fine-gained data sharing for cloud storage.

REFERENCES

Secure Outsourced Data Transfer with Integrity Verification in Cloud Storage

Jianbing Ni, Xiaodong Lin, Kuan Zhang, Yong Yu, and Xuemin (Sherman) Shen

Response of Comments

According to the comments of reviewers, we did the corresponding revisions on the manuscript as follows:

Comment 1: One weakness may be that the storage efficiency is not high since the outsourced data are encrypted by using asymmetric key encryption scheme.

Revision 1: The proposed scheme is not so efficient since the asymmetric key encryption is used to encrypt the outsourced data. How to improve the efficiency of the scheme is the work we are going to focus on. Currently, we have come up with an idea that the Merkle Hash Function can be used to achieve the data erasure verification. With the Merkle hash function, the secure data transfer scheme can support the data migration for both plaintexts and ciphertexts.

Comment 2: In the performance evaluation part, the authors may compare the SODT scheme with other schemes of the cloud data migration.

Revision 2: That is a good suggestion. But there is no such kind of scheme exist, so we do not show the comparison.

Comment 3: There are some typos in the paper, such as “the mathematical hard problems”–“mathematically...” “the cloud can preform secure data...” in the abstract part. Please number the theorem properly from Theorem 1.

Revision 3: We have corrected these typos.