Weighted Two-Levels Secret Sharing Scheme for Multi-Clouds Data Storage with Increased Reliability

Vanessa Miranda-López[1], Andrei Tchernykh[1,2,3]
[1] CICESE Research Center, Ensenada, Mexico
[2] South Ural State University, Chelyabinsk, Russia
[3] Ivannikov Institute for System Programming, Moscow, Russia,
{miranda, chernykh}@cicese.edu.mx,

Mikhail Babenko, Viktor Kuchukov, Maxim Deryabin,
Elena Golimblevskaia, Egor Shiryaev
North-Caucasus Federal University, Stavropol, Russia
{mgbabenko, vkuchukov, maderiabin}@ncfu.ru,
elena.golimblevskaya@gmail.com, ea_or@list.ru

Arutyun Avetisyan
Ivannikov Institute for System Programming,
Moscow, Russia Moscow, Russia
arut@ispras.ru

Raul Rivera-Rodriguez
CICESE Research Center,
Ensenada, Mexico,
rrivera@cicese.mx

Gleb Radchenko
South Ural State University
Chelyabinsk, Russia
gleb.radchenko@susu.ru

El-Ghazali Talbi
University of Lille 1 and
INRIA, Lille, France
el-ghazali.talbi@univ-lille.fr

***Abstract*— Cloud storages delivered as services are made available to the general public. As cloud storage provider prices become low, they have moved into the mainstream of storage technology. However, there are various factors that cause many potential users do not use it intensively. There exist high risks for confidentiality, integrity, and availability violation associated with the loss of information, denial of access, technical failures, etc. In this article, we propose a two-level secret sharing scheme (TL-SSS) based on a residue number system (RNS) for a configurable, reliable, and secure distributed data storage. RNS moduli of a special type increase the reliability of the data storage system and reduce the computational complexity of the data encoding and decoding from linear-logarithmic to linear. TL-SSS is the weighted data access structure. It creates and distributes data shares according to the characteristics of the cloud storages under various scenarios. We provide a solution that improves system reliability without reduction of the security level. In contrast to classical solutions, it can restore the data with less available shares than the state-of-the-art approaches.**

Keywords—cloud storage, reliability, residue number system, secret sharing scheme, uncertainty

I. INTRODUCTION

Cloud computing provides inexpensive and scalable resources and has gained widespread use for distributed computing. Scientific applications typically use resources not only for computing but also for data storage [1]. A large amount of data are generated during data analysis as intermediate or final results [2, 3]. The size of the files can reach several gigabytes or even terabytes. According to [4], the amount of scientific data is doubled each year.

A secure and fault-tolerant multi-cloud storage has to prevent information from unauthorized access, use, disclosure, disruption, modification, etc. Its confidentiality, integrity, and availability have to be preserved even in the presence of failures, deliberate, as well as accidental threats. To this end, Residue Number System (RNS), data encryption systems, homomorphic encryption, error correction codes, secured sharing schemes, etc. are widely used (see Section II). However, the use of homomorphic ciphers and error correction codes lead to a significant increase of the transferred, processed, and stored data.

In this paper, we propose an algorithm of the data encryption and error correction based on multilevel RNS with the diagonal function and recursive pairing method. We show how to select the moduli for supporting the asymptotic perfection. We also study the efficiency of addition and multiplication operations in RNS. We demonstrate that RNS with moduli of a special type reaches high efficiency of data encryption/decryption and keeps an appropriate level of storage security. The special moduli are widely used because they do not require complex operations, such as evaluation of multiplicative inverses, multiplication, etc. to convert RNS residues to/from positional representation [7].

An important operation for calculation of RNS residues is division [8]. However, it is a computationally expensive operation. The complexity of this operation during conversion from positional notation to RNS depends on the moduli-set. In [9], it is showed that $2^n - 1, 2^n, 2^{n+1} - 1$ moduli provide a high-performance system for RNS encoding and decoding. To build a storage system with high-performance and high transmission rate, we use a secret sharing scheme with two-level RNS on moduli of a special type $2^n - 1, 2^n, 2^{n+1} - 1$. It converts the data to the set of small shares and distributes them between cloud storages according to their characteristics to improve reliability.

This paper is organized as follows. Section II reviews distributed storage systems for clouds under uncertainty of errors, falsifications, loss of information, and technical failures. Section III discusses one level and two-level residue number systems and their properties. Section IV presents our data storage model. Section V analyzes data recovery and data redundancy degradations. Section VI focuses on modeling the proposed scheme. The conclusions and future work are discussed in the last Section VII.

II. RELATED WORK

One of the key issues of cloud storage is the fast recovery in case of critical failures that cannot be predicted or anticipated in advance [5].

Reliability is often associated with availability. Reliability, in particular, refers to how much you can rely on the ability of the system to protect the integrity of the data and perform its transactions in the face of uncertainty of system failures [10]. Availability is the proportion of time a system is in a functioning condition associated with system instability, data corruption, denial of access, etc.

To a great extent, the reliability of cloud storage depends on several aspects including the quality and stability of resource management [11]. To increase reliability, many approaches are used: data replication [12], Secret Sharing Schemes (SSS) [6], Redundant Residue Number System (RRNS) [5, 13], Erasure Codes (EC) [14], Regenerating Codes (RC) [15], Homomorphic Encryption (HE) [6, 16], computationally complex positional codes [8], etc.

The use of replication leads to high redundancy and cost. The positional codes lack control over the execution of arithmetic operations. Modular arithmetic codes provide in a single mechanism the reliability, security, and error correction. The latter feature is widely used to increase the fault tolerance of cloud data storages.

III. Two-Level Residue Number System and its Properties

RNS represents original numbers as residues with respect to a moduli set. Let us introduce the following notations.

D	original data
D_n	n lower bits of the number D
MT	pseudo-random sequence Mersenne Twister
S	secret data
n_1	number of moduli on the first level
$\mathrm{k}_1 \le \mathrm{n}_1$	threshold value for the secret sharing scheme
$\mathrm{r}_1 = \mathrm{n}_1 - \mathrm{k}_1$	number of control (redundant) RRNS moduli
$\mathrm{p}_{1,\mathrm{i}}$	i-th RNS modulus on the first level
$P = \prod_{i=1}^{k} p_{1,i}$	dynamic range of RRNS and $S \in [0, P)$
$S_i = \lvert S \rvert_{p_{1,i}}$	remainder of the division S by module $p_{1,i}$

$p_{1,1}, p_{1,2}, \dots, p_{1,n}$ are pairwise coprime numbers used as moduli set of RNS. It is known from the number theory that if the moduli $p_{1,i}$ are coprime then the representation of the number $S \xrightarrow{RNS} (S_1, S_2, \dots, S_n)$ is unique for all the numbers that satisfy a condition $P > \mathrm{S}$.

The constructive version of the Chinese Remainder Theorem (CRT) gives a method to recover S from RNS representation. In RRNS settings (k_1, n_1), S can be recovered from any k_1 remainders from n_1.

To guarantee the required dynamic range, we can use either a large number of small moduli or several large moduli. For small moduli, the operation of converting numbers from RNS to a binary number system is more computationally complex. It is desirable that the selected moduli have effective software and hardware implementations of the basic modular operations.

There are three classes of moduli [3]:

- Even moduli, which are powers of 2, i.e., $p = 2^n$.
- Odd moduli of the form $p = 2^n - 1$, which have the lower computational complexity among general odd moduli for data encoding and decoding.
- Odd moduli of the form $p = 2^n + 1$.

The multi-level RNS is a recursive extension of the classical one-level RNS.

On the first level, n_1 moduli $p_{1,1}, p_{1,2}, \dots, p_{1,n_1}$ are used to calculate $S_1, S_2, \dots, S_{n_1}$. On the second level, each S_i is transformed to the set of residuals $S_{i,j} = \lvert S_i \rvert_{p_{2,i,j}}$ by its own moduli set $p_{2,i,1}, p_{2,i,2}, \dots, p_{2,i,n_{2,i}}$ (Fig.1).

Let us introduce the following notations for two-level RNS.

$n_{2,i}$	number of moduli $p_{2,i,1}, p_{2,i,2}, \dots, p_{2,i,n_{2,i}}$ used to calculate S_i for each $i = \overline{1, n_1}$.
$\mathrm{k}_{2,\mathrm{i}} \le \mathrm{n}_{2,\mathrm{i}}$	threshold value for secret sharing scheme used in the representation S_i
$\mathrm{r}_{2,\mathrm{i}} = \mathrm{n}_{2,\mathrm{i}} - \mathrm{k}_{2,\mathrm{i}}$	number of control (redundant) RRNS moduli used in the representation S_i
$p_{2,i,j}$	RRNS modulus used in the representation S_i for each $i = \overline{1, n_1}$ and j= $\overline{1, n_{2,i}}$.
$M_i = \prod_{j=1}^{k_{2,i}} p_{2,i,j} \ge p_{1,i}$	dynamic range of RRNS definite moduli set $\{p_{2,1}, p_{2,2}, \dots, p_{2,n_i}\}$ and $S_i \in [0, p_{1,i})$ for each $i = \overline{1, n_1}$
$S_{i,j} = \lvert S_i \rvert_{p_{2,i,j}}$	remainder of the division S_i by module $p_{2,i,j}$ for each $i = \overline{1, n_1}$ and j $= \overline{1, n_{2,i}}$.

S_i satisfies the condition $S_i < p_{1,i}$, for all $i = \overline{1, n_1}$. From the CRT, it follows that for a one-to-one mapping between $S_i \in [0,\ p_{1,i})$ and $\left(S_{i,1}, S_{i,2}, \dots, S_{i,n_{2,i}}\right)$, it is necessary and sufficient that $M_i \ge p_{1,i}$ for each $i = \overline{1, n_1}$.

$$\mathrm{M}_\mathrm{i} = \prod_{\mathrm{j}=1}^{\mathrm{k}_{2,\mathrm{i}}} \mathrm{p}_{2,\mathrm{i,j}} \ge p_{1,i}$$

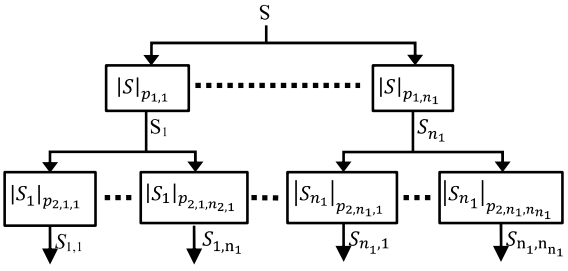

Figure 1. Data encoding scheme to two-level RNS

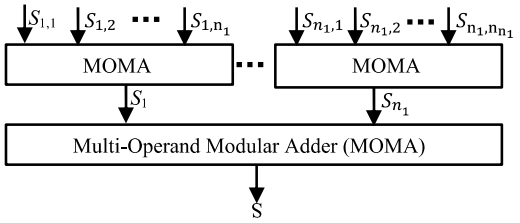

Figure 2. Data decoding scheme from two-level RNS

Fig. 2 shows a data decoding scheme. First, S_i for all $i = \overline{1, n_1}$ are restored from the corresponding $S_{i,j}$, then S is restored from $S_1, S_2, \dots, S_{n_1}$.

Multi-Operand Modulo Addition (MOMA) is an algorithmic primitive that accepts n_1 operands $S_1, S_2, \dots, S_{n_1}$, with $0 \le S_i < p_{1,i}$ for each $i = \overline{1, n_1}$ and computes the residue of their sum taken modulo P. That is, it computes $S = \left| w_i S_1 + w_2 S_2 + \cdots + w_{n_1} S_{n_1} \right|_P$, where $w_i = P_i \cdot |P_i^{-1}|_{p_{1,i}}$ and $P_i = P/p_{1,i}$ for all $= \overline{1, n_1}$.

In TL-RRNS, the reliability of the system depends on the parameters $k_1, n_1, k_{2,1}, n_{2,1}, k_{2,2}, n_{2,2}, \dots, k_{2,n_1}, n_{2,n_1}$. (see Section VI)

IV. Data Storage Model

One of the advantages of RNS for building reliable data storage systems is the ability to detect and correct errors as an error correction code. On the other hand, the mapping data to the RNS allows ensuring data security and secret sharing.

However, one of the drawbacks of secret-sharing schemes using RNS is the low data encoding and decoding rate. To increase the speed of encoding and decoding data, we use a special type of moduli that reduce the computational complexity from linear-logarithmic to linear.

A deterrent factor in choosing a special type of RNS modules is the requirement of their mutual simplicity. It limits the number of special type modules. The most efficient moduli in terms of a software implementation of algorithms for data encoding and decoding are moduli of two types: even and odd.

- Even moduli, which are powers of 2, i.e. $p = 2^n$, can be used only once in the RNS moduli set. Moreover, the use of the module of the form $p = 2^n$ allows an attacker to know a part of confidential data without decryption. For example, let $S = 101101110_2$ and $p = 2^3$, then the attacker can obtain last three bits 110 in the binary representation, $|S|_p = |101101110|_{2^3} = 110$.
- Odd moduli of the form $p = 2^n - 1$ have the lower computational complexity among general odd moduli for data encoding and decoding.

The requirement of mutual simplicity of the moduli does not allow to select moduli of a special type $p = 2^n - 1$, where n are successive numbers of the natural series. For example, let us consider nine moduli: $2^3 - 1, 2^4 - 1, 2^5 - 1, 2^7 - 1, 2^{11} - 1, 2^{13} - 1, 2^{17} - 1, 2^{19} - 1, 2^{23} - 1$. When we divide an original data on moduli $2^3 - 1$ and $2^{23} - 1$, the first remainder provides $\frac{23}{3} \approx 7.7$ times less information about the original data than the second one.

From all the above, it follows that the construction of a distributed data storage scheme using a single-level RNS built on modules of the form $\{2^{n_1}, 2^{n_2} - 1, 2^{n_3} - 1, \dots, 2^{n_t} - 1\}$ is not advisable, since it leads to a security imbalance. The pursuit of the possibility of moduli $\{2^n - 1, 2^n, 2^{n+1} - 1\}$ leads to the idea of building RNS in two levels.

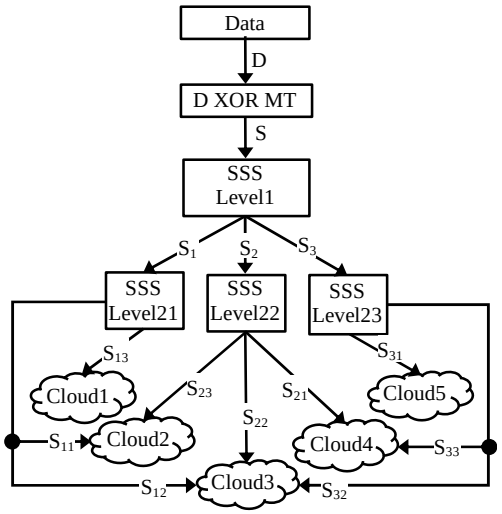

Figure 3. Data storage model (2,3)-(2,3), (2,3), (2,3)

Fig.3 shows the fundamental scheme of the storage model based on two-level RRNS with $n_1 = 3$, $n_{2,1} = n_{2,2} = n_{2,3} = 3$.

1. The data is encrypted by a binary XOR operation with the key generated by the Mersenne Twister algorithm.
2. RNS with the settings (k_1, n_1) is applied on the first level to part encrypted data to $n_1 = 3$ shares with k=2, where $p_{1,1} = 2^n - 1$, $p_{1,2} = 2^n$, $p_{1,3} = 2^{n+1} - 1$, and n depend on the size of D (see Algorithm 2 bellow).
3. RNS with the settings $(k_{2,1}, n_{2,1})$, $(k_{2,2}, n_{2,2})$, $(k_{2,3}, n_{2,3})$ is applied on the second level to part each encrypted data S_i to $n_{2,1} = n_{2,2} = n_{2,3} = 3$ shares with $k_{2,1} = k_{2,2} = k_{2,3} = 2$, where $p_{2,1,1} = 2^{m_{12}} - 1$, $p_{2,1,2} = 2^{m_{12}}$, $p_{2,1,3} = 2^{m_{12}+1} - 1$, $p_{2,2,1} = 2^{m_{12}} - 1$, $p_{2,2,2} = 2^{m_{12}}$, $p_{2,2,3} = 2^{m_{12}+1} - 1$, and $p_{2,3,1} = 2^{m_3} - 1$, $p_{2,3,2} = 2^{m_3}$, $p_{2,3,3} = 2^{m_3+1} - 1$. m_{12} and m_3 depend on n.
4. $n_{2,1} + n_{2,2} + n_{2,3} = 9$ shares are stored in $m = 5$ cloud storages according to the weighted SSS (1, 2, 3, 2, 1) that corresponds the number of shares allocated to the first, second, and so on storages.

Since the modulus $p = 2^n$ is used, it is necessary to apply an additional algorithm for data noising to increase the security level. Mersenne Twister (MT) algorithm is suitable for this purpose, due to it has a high rate of pseudo-random sequence generation with good statistical property: a 623-dimensionally equidistributed uniform pseudorandom distribution during a large period.

Algorithm 1. Data encryption algorithm using MT.
Input: D - input data, MT - pseudo-random sequence
Output: S – encrypted data
1. $key = D$ ShiftRigth n
2. $S = D$
3. For $i = 0$ to $n - 1$ do: $S[i] = D[i]\ XOR\ MT[\text{key} + i]$
4. **return** S

Since D can be represented as $D = key \cdot 2^n + D_n$, $S = D\ XOR\ MT[key : key + n - 1]$, where $MT[key : key +$

$n-1]$ is the substring of bits between bit index key and bit index $key + n - 1$ in the pseudo-random sequence.

Data encryption based on MT is presented in Algorithm 1. To calculate shares S_1, S_2, S_3, S_{11}, S_{12}, S_{13}, S_{21}, S_{22}, S_{23}, S_{31}, S_{32}, S_{33}, it is proposed the following Algorithm 2.

Algorithm 2. Data coding algorithm based on weighted two-levels RNS.

Input: S - Encrypted data

Output: $S_{11}, S_{12}, S_{13}, S_{21}, S_{22}, S_{23}, S_{31}, S_{32}, S_{33}$ – RNS coded shares

1. $L = NumberOfBits(S)$
2. $n = L\ ShiftRigth\ 1$
3. If $|L|_2 == 1$ then $n = n + 1$
4. $S_1 = |S|_{2^n-1}, S_2 = |S|_{2^n}, S_3 = |S|_{2^{n+1}-1}$
5. $m_{12} = (\mathrm{n} + 1)\ ShiftRigth\ 1,$
6. $m_3 = 1 + n\ \ ShiftRigth\ 1$
7. $S_{11} = |S_1|_{2^{m_{12}}-1}, S_{12} = |S_1|_{2^{m_{12}}}, S_{13} = |S_1|_{2^{m_{12}+1}-1}$
8. $S_{21} = |S_2|_{2^{m_{12}}-1}, S_{22} = |S_2|_{2^{m_{12}}}, S_{23} = |S_2|_{2^{m_{12}+1}-1}$
9. $S_{31} = |S_3|_{2^{m_3}-1}, S_{32} = |S_3|_{2^{m_3}}, S_{33} = |S_3|_{2^{m_3+1}-1}$
10. **return** $S_{11}, S_{12}, S_{13}, S_{21}, S_{22}, S_{23}, S_{31}, S_{32}, S_{33}$

From Algorithm 2, it follows that $S \in [0, 2^{2n} - 1]$ when $|L|_2 = 0$ and $S \in [0, 2^{2n-1} - 1]$ when $|L|_2 = 1$. From line 3 of Algorithm 2, it follows that $n = L - \lfloor L/2 \rfloor$.

Let us consider an example of Algorithm 2 application.

Example 1. Let $S = 5850495393454642923$ then $L = NumberOfBits(S) = 63$ bits, then $n = L - \lfloor L/2 \rfloor = 32$.

Let us compute shares S_1, S_2, S_3.

$S_1 = |S|_{2^{32}-1} = 2015197563, S_2 = |S|_{2^{32}} = 653022955,$

$S_3 = |S|_{2^{33}-1} = 1334110259$

We calculate the values of degrees m_{12} and m_3 for the second-level SSS. We have

$m_{12} = (\mathrm{n} + 1)\ ShiftRigth\ 1 = 16,$

$m_3 = 1 + n\ \ ShiftRigth\ 1\ = 17.$

Shares are calculated in three parallel processes.

The first process. The input of the SSS level 21 is the value $S_1 = 2015197563$. The shares of S_1 are calculated:

$S_{11} = |S_1|_{2^{m_{12}}-1} = |S_1|_{2^{16}-1} = 61848,$

$S_{12} = |S_1|_{2^{m_{12}}} = |S_1|_{2^{16}} = 31099,$

$S_{13} = |S_1|_{2^{m_{12}+1}-1} = |S_1|_{2^{17}-1} = 112009.$

The second process. The input of the SSS level 22 is the value $S_2 = 653022955$. The shares of S_2 are calculated:

$S_{21} = |S_2|_{2^{m_{12}}-1} = |S_{12}|_{2^{16}-1} = 32215,$

$S_{22} = |S_2|_{2^{m_{12}}} = |S_2|_{2^{16}} = 22251,$

$S_{23} = |S_2|_{2^{m_{12}+1}-1} = |S_2|_{2^{17}-1} = 27233.$

The third process. The input of the SSS level 23 is the value $S_3 = 1334110259$. The shares of S_3 are calculated:

$S_{31} = |S_3|_{2^{m_3}-1} = |S_3|_{2^{17}-1} = 69621,$

$S_{32} = |S_3|_{2^{m_3}} = |S_3|_{2^{17}} = 59443,$

$S_{33} = |S_3|_{2^{m_3+1}-1} = |S_3|_{2^{18}-1} = 64532.$

Thus, the result of the algorithm is nine coded shares of the initial data. We distribute them between five cloud storages according to the weighted SSS (1, 2, 3, 2, 1) as shown in Fig. 1. Clouds 1 and Cloud 5 have one share, Cloud 2 and Cloud 4 have 2 shares, and Cloud 3 has three shares. The properties of such a scheme are discussed in Section V.

V. Data Recovery

A. Case 1

Let us assume that two clouds from five are available: 1 and 3. Hence, we know $S_{13}, S_{12}, S_{22}, S_{32}$. Using the values of S_{13} and S_{12}, we restore the value of S_1. Thus, the values of S can be represented as $S = z(2^n - 1) + S_1$, where

$$z \le \begin{cases} 2^n + 1, if\ |L|_2 = 0 \\ 2^{n-1}, if\ |L|_2 = 1 \end{cases}$$

Rewrite S in another form

$$S = z(2^n - 1) + S_1 =$$
$$= \frac{z - z_0}{2}(2^{n+1} - 1) + S_1 - \frac{z + z_0}{2} + z_0 2^n, \quad (1)$$

where z_0 is the low-order bit of the word z.

From Eq. (1), it follows that

$$S_{22} = |S_2|_{2^{m_{12}}} = |S_1 - z|_{2^{m_{12}}} \quad (2)$$

$$S_{32} = |S_3|_{2^{m_3}} = \left|S_1 - \frac{z + z_0}{2}\right|_{2^{m_3}} \quad (3)$$

From the Eq. (2), we calculate the last m_{12} bits

$$|z|_{2^{m_{12}}} = |S_1 - S_{22}|_{2^{m_{12}}} \quad (4)$$

Using the Eq. (3) and (4), we calculate the values of the bit $|z|_{2^{m_3}}$, if $m_3 = m_{12}$, and the values of the bits m_3 and m_{12}, if $m_3 = m_{12} + 1$. Thus, knowing the four parts, we can recover only $n + m_3 + 1$ bits from $2n - 1$ bits of information. Therefore, we can calculate only $3/4$ of the cipher of the text and $1/4$ remains unknown.

When Cloud 3 and Cloud 5 are available, the recovery is proved in the same way as the above. Therefore, if we only have shares stored in Clouds 1 and 3, we can restore the original data. However, if we only have shares stored in Clouds 2, 4, or in Clouds 3, 5, we cannot restore the original data.

B. Case 1I

Let us assume that two clouds are available: Cloud 2 and Cloud 4. Therefore, $S_{11}, S_{21}, S_{23}, S_{33}$ are known. According to the CRT, we can restore the value S_2 based on S_{21}, S_{23}.

Therefore, S can be represented as $S = z2^n + S_2$, where

$$z \le \begin{cases} 2^n - 1, if\ |L|_2 = 0 \\ 2^{n-1} - 1, if\ |L|_2 = 1 \end{cases}$$

Let us rewrite S as follows

$$S = z2^n + S_2 = z(2^n - 1) + S_2 + z = \quad (5)$$

$$= \frac{z - z_0}{2}(2^{n+1} - 1) + S_2 + \frac{z - z_0}{2} + z_0 2^n$$

Thus, from Eq. (5), it follows that

$$S_1 = |S_2 + z|_{2^n - 1} \tag{6}$$

$$S_3 = \left| S_2 + \frac{z - z_0}{2} + z_0 2^n \right|_{2^{n+1}-1} \tag{7}$$

Let $z_0 = 0$, then the Eq. (7) takes the form

$$S_3 = \left| S_2 + \frac{z}{2} \right|_{2^{n+1}-1}$$

Since $S_2 < 2^n$ and $\frac{z}{2} < 2^{n-1}$, then $S_2 + \frac{z}{2} < 2^n + 2^{n-1} < 2^{n+1} - 1$. Therefore,

$$S_3 = S_2 + \frac{z}{2} \text{ and } z = 2 \cdot (S_3 - S_2) \tag{8}$$

Let $z_0 = 1$, then Eq. (7) takes the form

$$S_3 = \left| S_2 + \frac{z - 1}{2} + 2^n \right|_{2^{n+1}-1}$$

Therefore,

$$S_3 = \begin{cases} S_2 + \dfrac{z - 1}{2} + 2^n, & \text{if } \Delta < 2^{n+1} - 1 \\ S_2 + \dfrac{z \mp 1}{2} - 2^n, & \text{else} \end{cases}$$

where $\Delta = S_2 + \frac{z-1}{2} + 2^n$.

We calculate the value

$$z = \begin{cases} 2S_3 - 2S_2 - 2^{n+1} + 1 & if\ \widetilde{\Delta} \geq 0 \\ 2S_3 - 2S_2 + 2^{n+1} - 1 & if\ \widetilde{\Delta} < 0, \end{cases} \tag{9}$$

where $\widetilde{\Delta} = 2S_3 - 2S_2 - 2^{n+1} + 1$.

If n is even, $n = 2m_{12}$, therefore $(2^{m_{12}} - 1)|(2^n - 1)$ and the expression

$$S_{11} = |S_1|_{2^{m_{12}}-1} = ||S_2 + z|_{2^n-1}|_{2^{m_{12}}-1} = |S_2 + z|_{2^{m_{12}}-1},$$

It means that

$$|z|_{2^{m_{12}}-1} = |S_{11} - S_2|_{2^{m_{12}}-1} \tag{10}$$

If m_{12} is even, then $3|\gcd(2^{m_{12}} - 1, 2^{m_3+1} - 1)$. Let

$$|2S_{33} - 2S_2|_3 = x \tag{11}$$

then using Eq. (9), we have

$$|2S_3 - 2S_2 - 2^{n+1} + 1|_3 = |x + 2|_3 \tag{12}$$

$$|2S_3 - 2S_2 + 2^{n+1} - 1|_3 = |x + 1|_3 \tag{13}$$

By checking the conditions $|z|_3 = |S_2 + z|_3$ from Eq. (10) and from Eq. (11) - (13), we calculate the value $|z|_3 = (2^{m_3+1} - 1)$. Using the CRT, we can restore the value of z and find the value of S if we know the values of $|z|_{2^{m_3+1}-1}$ and $|z|_{2^{m_{12}}-1}$. This possibility outperforms the traditional one-level RNS, where data cannot be restored.

If $n = 2m_{12}$ and m_{12} is odd, then $\gcd(2^{m_{12}} - 1, 2^{m_3+1} - 1) = 1$. Using Eq. (10), we calculate the value $|z|_{2^{m_{12}}-1}$.

Let $m_{12} = m$, then $m_3 = m + 1$. Define the parameters of the RNS moduli set is $(2^{m_{12}} - 1, 2^{m_3+1} - 1)$:

$$P = (2^m - 1)(2^{m+2} - 1),\ P_1 = 2^{m+2} - 1, P_2 = 2^m - 1$$

$$B_1 = \left| \frac{2^{m+1} - 1}{3} \right|_{2^m - 1} \cdot (2^{m+2} - 1) \tag{14}$$

Since $3|(2^{m+1} - 1)$, then Eq. (14) can be rewritten in the form:

$$B_1 = \frac{2^{m+1} - 1}{3} \cdot (2^{m+2} - 1) \tag{15}$$

$$B_2 = \left| \frac{4 \cdot (2^{m+3} - 1)}{-3} \right|_{2^{m+2}-1} \cdot (2^m - 1) \tag{16}$$

Since $3|(2^{m+3} - 1)$, then Eq. (16) takes the form

$$B_2 = -\frac{4}{3}(2^{m+3} - 1) \cdot (2^m - 1) \tag{17}$$

Considering that calculating the value of z according to the CRT is performed with modulo P, we have

$$\begin{aligned} B_2 &= 3 \cdot P - \frac{4}{3}(2^{m+3} - 1) \cdot (2^m - 1) \\ &= \frac{1}{3}(2^m - 1) \cdot (2^{m+2} - 5) \end{aligned} \tag{18}$$

Let $|S_{11} - S_2|_{2^m - 1} = a_1$ and $|z|_{2^{m+2}-1} = |2S_{33} - 2S_2|_{2^{m+2}-1} = a_2$, then

$$|2S_{33} - 2S_2 - 2^{n+1} + 1|_{2^{m+2}-1} = \begin{cases} a_2 - 2^{m-1} + 1, & \text{if } a_2 \geq 2^{m-1} - 1 \\ a_2 + 7 \cdot 2^{m-1}, & \text{else} \end{cases}$$

$$|2S_{33} - 2S_2 + 2^{n+1} - 1|_{2^{m+2}-1} = \begin{cases} a_2 + 2^{m-1} - 1, & \text{if } a_2 \geq 7 \cdot 2^{m-1} \\ a_2 - 7 \cdot 2^{m-1}, & \text{else} \end{cases}$$

1. If $a_2 < 2^{m-1} - 1$, then let $|a_1 B_1 + a_2 B_2|_P = y$, then

$$z = \begin{bmatrix} y \\ \left| y + \frac{4}{3}(2^{m-1} - 1) \cdot (2^m - 1) \right|_P \\ \left| y + \frac{1}{3}(2^m - 1) \cdot (2^{m+1} - 1) \right|_P \end{bmatrix}$$

Since $3|(2^{m-1} - 1)$, then $2^{2m-1} < \frac{4}{3}(2^{m-1} - 1)(2^m - 1) < 2^{2m}$.

Considering that $3|(2^{m-1} - 1)$, we get $2^{2m-1} < \frac{1}{3}(2^m - 1)(2^{m+1} - 1) < 2^{2m}$. If $0 < y < 2^{2m-2}$, then

$$2^{n-1} < 2^{2m-1} < y + \frac{4}{3}(2^{m-1} - 1)(2^m - 1) < P,$$

$$2^{n-1} < 2^{2m-1} < y + \frac{1}{3}(2^m - 1)(2^{m+1} - 1) < P.$$

Since $\left(y + \frac{1}{3}(2^m - 1)(2^{m+1} - 1)\right) - \left(y + \frac{4}{3}(2^{m-1} - 1)(2^m - 1)\right) = 2^m - 1$, then $0 \geq y + \frac{4}{3}(2^{m-1} - 1)(2^m - 1) < 2^{2m-2} - 2^m + 1$, then two z values are possible.

2. If $2^{m-1}-1 \geq a_2 \geq 7\cdot 2^{m-1}$, then

$$z = \begin{cases} y \\ \left|y-\frac{4}{3}(2^{m-1}-1)\cdot(2^{m}-1)\right|_P \\ \left|y+\frac{1}{3}(2^{m}-1)\cdot(2^{m+1}-1)\right|_P \end{cases}$$

If $0 < y < 2^{2m-2}$, then $2^{n-1} < 2^{2m+1} < P + y - \frac{1}{3}(2^{m-1}-1)(2^m-1) < P$, $2^{n-1} < 2^{2m-1} < y + \frac{1}{3}(2^m - 1)(2^{m+1}-1) < 2^{2m} + 2^{2m-2} < P$.

Since $\left(y+\frac{1}{3}(2^m-1)(2^{m+1}-1)\right) - \left(y-\frac{4}{3}(2^{m-1}-1)(2^m-1)\right) = \frac{2(2^m-1)(2^{m+1}-1)}{3} > 2^{2m-2}$, then we can restore the value z unequivocally.

3. If $a_2 > 7\cdot 2^{m-1}$, then

$$z = \begin{cases} y \\ \left|y-\frac{4}{3}(2^{m-1}-1)\cdot(2^{m}-1)\right|_P \\ \left|y+\frac{4}{3}(2^{m-1}-1)\cdot(2^{m}-1)\right|_P \end{cases}$$

If $0 < y < 2^{2m-2}$, then

$$2^{n-1} < 2^{2m+1} < P + y - \frac{4}{3}(2^{m-1}-1)(2^m-1) < P,$$

$$2^{n-1} < 2^{2m-1} < y + \frac{4}{3}(2^{m-1}-1)(2^m-1) < P.$$

Since $\left(y+\frac{4}{3}(2^{m-1}-1)(2^m-1)\right) - \left(y-\frac{4}{3}(2^{m-1}-1)(2^m-1)\right) = \frac{8}{3}(2^{m-1}-1)(2^m-1) > 2^{2m-2}$, then we can restore value z unequivocally.

We calculate three possible values of z using Eq (8) and (9). We restore z using the CRT and check that the condition $z < 2^{n-1}$ is satisfied and determine the desired value of z.

For n is odd, $n+1 = 2m$, therefore $(2^{m_3}-1)|(2^{n+1}-1)$ and the expression $S_{33} = |S_3|_{2^{m_3}-1} = \left|\left|S_2 + \frac{z-z_0}{2} + z_0 2^n\right|_{2^{n+1}-1}\right|_{2^{m_3}-1} = \left|S_2 + \frac{z-z_0}{2} + z_0 2^{m_3-1}\right|_{2^{m_3}-1}$, therefore,

$$\left|\frac{z-z_0}{2}\right|_{2^{m_3}-1} = |S_{33} - S_2 - z_0 2^{m_3-1}|_{2^{m_3-1}} \qquad (19)$$

From the Eq. (19), we get

$$S_{11} = \begin{cases} |S_2 + z|_{2^{m_{12}}-1} \\ |S_2 + z - 2^{m_{12}-1} + 1|_{2^{m_{12}}-1} \end{cases} \qquad (20)$$

Expressing from Eq. (20) the values $|z|_{2^{m_{12}}-1}$, we get

$$|z|_{2^{m_{12}}-1} = \begin{cases} |S_{11} + S_2|_{2^{m_{12}}-1} \\ |S_{11} - S_2 - z + 2^{m_{12}} - 1|_{2^{m_{12}}-1} \end{cases}$$

Using the CRT, we reconstruct four possible values of z on the second level. After that, we reconstruct the data by z. Therefore, using shares of Cloud 2 and Cloud 4, we restore the original data. The traditional one-level RNS cannot restore the data in this case.

C. Examples

To show the applicability of the proposed scheme, we present and discuss two examples.

Example 2. Let us show the recovery process when Cloud 1, Cloud 2, Cloud 4, Cloud 5 are available, hence, S_{11}, S_{13}, $S_{21}, S_{23}, S_{31}, S_{33}$ are known. In the beginning, two processes are launched: restoring S_1 and S_3.As a result of the first process, the share S_1 is restored by the Eq. (21).

$$S_1 = |(S_{11} - S_{13})\cdot(2^{m_{12}+1}-1) + S_{13}|_{((2^{m_{12}}-1)\cdot(2^{m_{12}+1}-1))}, \qquad (21)$$

where $S_{11} = 61848, S_{13} = 112009, m_{12} = 16$, $S_1 = |(61848 - 112009)\cdot(2^{17}-1) + 112009|_{(2^{16}-1)\cdot(2^{17}-1)} = 2015197563$.

The second process is to restore the value of the projection S_3 by Eq. (22).

$$S_3 = |(S_{31} - S_{33})\cdot(2^{m_3+1}-1) + S_{33}|_{(2^{m_3}-1)\cdot(2^{m_3+1}-1)} \qquad (22)$$

$S_{31} = 69621$, $S_{33} = 64532$, $m_3 = 17$, $S_3 = |(69621 - 64532)\cdot(2^{18}-1) + 64532|_{(2^{17}-1)\cdot(2^{18}-1)} = 1334110259$.

Let us calculate the value of S based on the obtained values of S_1 and S_3 using the Eq. (23).

$$S = |(S_1 - S_3)\cdot(2^{n+1}-1) + S_3|_{(2^n-1)\cdot(2^{n+1}-1)} \qquad (23)$$

where $S_1 = 2015197563, S_3 = 1334110259, n = 32$. $S = |(2015197563 - 1334110259)\cdot(2^{33}-1) + 1334110259|_{(2^{32}-1)\cdot(2^{33}-1)} = 5850495393454642923$. As a result, the algorithm restores the value of S.

Now, let us give examples of the algorithm at the stage of information recovery, with the availability of various storages.

Example 3. Let us consider the case when there is access to two cloud storages Cloud 2 and Cloud 4, i.e. we have access to four parts of the projections: S_{11}, S_{21}, S_{23} и S_{33}, where $S_{11} = 48427$, $S_{21} = 27233$, $S_{23} = 24742$, $S_{33} = 164191$.

1. Using the values $S_{21} = 27233$, $S_{23} = 24742$, we calculate $S_2 = |(S_{21} - S_{23})(2^{18}-1) + S_{33}|_{(2^{16}-1)\cdot(2^{18}-1)} = 653022955$

2. Now, we calculate $|z|_{2^{17}-1} = |S_{11} - S_2|_{2^{17}-1} = 21194$

Let $z_0 = 0$, then

$$|z|_{2^{19}-1} = \begin{cases} |2\cdot(S_{33} - S_2)|_{2^{19}-1} = 281389 \\ |2\cdot(S_{33} - S_2 + 2^{16} - 1)|_{2^{19}-1} = 412459 \end{cases}$$

Let $z_0 = 1$, then

$$|z|_{2^{19}-1} = \begin{cases} |2\cdot(S_{33} - S_2 - 2^{15}) + 1|_{2^{19}-1} = 215854 \\ |2\cdot(S_{33} - S_2 + 2^{15} - 1) + 1|_{2^{19}-1} = 346924 \end{cases}$$

We compute the values $P = (2^{17}-1)\cdot(2^{19}-1)$, then $P_1 = 2^{19}-1$ and $P_2 = 2^{17}-1$. $k_1 = |P_1^{-1}|_{2^{17}}\cdot P_1 = 45812722347$, $k_2 = |P_2^{-1}|_{2^{19}-1}\cdot P_2 = 22906099031$.

We find the values of z:

$z \xrightarrow{RNS} (21194,281389)$, then $z = |a_1 \cdot k_1 + a_2 \cdot k_2|_P = 340543652 < 2^{33}$ satisfies the condition $z < 2^{33}$.

$z \xrightarrow{RNS} (21194,412459)$, then $z = |a_1 \cdot k_1 + a_2 \cdot k_2|_P = 46153397069 > 2^{33}$ does not satisfy the condition $z < 2^{33}$.

$z \xrightarrow{RNS} (21194,215854)$, then $z = |a_1 \cdot k_1 + a_2 \cdot k_2|_P = 11793527632 > 2^{33}$ does not satisfy the condition $z < 2^{33}$.

$z \xrightarrow{RNS} (21194,346924)$, then $z = |a_1 \cdot k_1 + a_2 \cdot k_2|_P = 57606381049 > 2^{33}$ does not satisfy the condition $z < 2^{33}$.

Therefore, $z = 340543652$. To recover data S, we have:

$S = z \cdot 2^{34} + S_2 = 5850495393454642923$.

D. Data Redundancy Degradation

Redundancy in big data storage is an important issue. In the worst case, the number of bits that need to be stored is equal to L. The number of bits for shares $S_{11}, S_{12}, S_{21}, S_{22}$ is m_{12} bits, for S_{13}, S_{23} is $m_{12} + 1$-bits, for S_{31}, S_{32} is m_3-bits and S_{33} is $m_3 + 1$. We calculate redundancy degradation as the ratio of the stored encoded data to the original data size minus 1:

$$R = \frac{6 \cdot m_{12} + 3 \cdot m_3 + 3}{L} - 1.$$

Four cases are possible.

Case 1. If $L = 4 \cdot k$, then $n = 2 \cdot k$, therefore $m_{12} = k$ and $m_3 = k + 1$, well then $6 \cdot m_{12} + 3 \cdot m_3 + 3 = 9 \cdot k + 6$ and $R = \frac{9 \cdot k + 6}{4k} - 1 = 1.25 + \frac{3}{2k}$.

Case 2. If $L = 4 \cdot k + 1$, then $n = 2 \cdot k + 1$, therefore $m_{12} = k + 1$ and $m_3 = k + 1$, well then $6 \cdot m_{12} + 3 \cdot m_3 + 3 = 9 \cdot k + 12$ and $R = \frac{9 \cdot k + 12}{4k + 1} - 1 = 1.25 + \frac{39}{16 \cdot k + 4}$.

Case 3. If $L = 4 \cdot k + 2$, then $n = 2 \cdot k + 1$, therefore $m_{12} = k + 1$ and $m_3 = k + 1$, well then $6 \cdot m_{12} + 3 \cdot m_3 + 3 = 9 \cdot k + 12$ and $R = \frac{9 \cdot k + 12}{4k + 2} - 1 = 1.25 + \frac{15}{8 \cdot k + 4}$.

Case 4. If $L = 4 \cdot k + 3$, then $n = 2 \cdot k + 2$, therefore $m_{12} = k + 1$ and $m_3 = k + 2$, well then $6 \cdot m_{12} + 3 \cdot m_3 + 3 = 9 \cdot k + 15$ and $R = \frac{9 \cdot k + 15}{4k + 3} - 1 = 1.25 + \frac{33}{16 \cdot k + 12}$.

In all four cases, $\lim_{k \to \infty} R = 1.25$. Therefore, the redundancy degradation asymptotically tends to 1.25 of the size of the input data.

E. Reliability Analysis

In the above examples, we compare classical two-level RNS and our scheme considering reliability. There are 382 combinations of the localization of four or more shares between five clouds needed for recovering. We show that the classical scheme can recover 346 from 382 cases, while our scheme can recover 382 from 382 cases, improving reliability on 10.4%. Schemes have the same encoding and decoding speeds (using moduli of a special type). Nevertheless, we increase reliability. When we compare schemes with the same reliability, for instance, with the single-level (4, 9) RRNS, we outperform it in the speed of the data encoding and decoding. In our scheme, the computational complexity of the algorithm for encoding and decoding data is linear, while in one-level RRNS is linear-logarithmic [5].

We search for a method to achieve higher speed with higher security and simplicity of moduli. Our two-level RNS gains these benefits. We have fewer moduli compared with one-level RNS keeping them small to provide a high level of security.

VI. MODELING

Let us introduce the following notations.

size(D)	Size of original data D,
T_E, T_D	Data encryption, decryption time,
t_{dow}, t_{up}	Downloading, uploading time from clouds
V_u, V_d	Uploading, downloading velocity

$$V_u = \frac{\text{size}(D)}{T_E + t_{up}}, V_d = \frac{\text{size}(D)}{T_D + t_{dow}}$$

The uploading velocity includes encoding and uploading to cloud storages. The downloading velocity includes downloading from cloud storages, decoding from RNS to binary. $t_{up} = \max\left(t_{up}(S_{ij})\right)$, $t_{dow} = \max_{S_{i,j} \in I}\left(t_{dow}(S_{i,j})\right)$ and I set of all possible chunks allows you to decode D.

We develop the experimental system on C# programming language. Experiments are performed on the PC with Intel Core i7-4790, 3.6GHz under Microsoft Window 10, and 100 Mbps internet connection. The experimental system includes six cloud storages: Mega, DropBox, YandexDisk, OneDrive, Box, and GoogleDrive.

Table I and Table II show the mean uploading and downloading speed obtained for data of 10 KB, 100 KB, 1 MB, 10 MB, and 100 MB. Times are measured every four hours during ten days obtaining 60 corresponding values.

TABLE I. AVERAGE UPLOADING SPEED VERSUS DATA SIZE (KB/S)

Cloud storage	10 KB	100 KB	1 MB	10 MB	100 MB
Mega	4.3	51.2	313.6	751.6	738.7
DropBox	3.6	28.8	219.9	631.9	1029.5
YandexDisk	**9.8**	**81.0**	153.2	373.3	593.6
OneDrive	2.7	60.2	271.9	562.0	1496.9
Box	2.8	26.3	204.8	**1137.7**	**1711.6**
GoogleDrive	4.5	46.2	**411.8**	1073.9	1443.2

TABLE II. AVERAGE DOWNLOADING SPEED VERSUS DATA SIZE (KB/S)

Cloud storage	10 KB	100 KB	1 MB	10 MB	100 MB
Mega	3.48	39.83	365.27	587.25	747.35
DropBox	7.46	41.54	465.44	654,.2	1226.40
YandexDisk	**10.36**	**123.44**	**1063.63**	**2123.94**	**2637,58**
OneDrive	2.69	41.49	433.86	1227.73	1210.19
Box	2.74	36.59	362.78	658.26	1014.50
GoogleDrive	3.08	52.07	511.97	2031.61	2192.93

Tables I, II and Table III show that in the worst case, the data encoding speed is $25.6/1711.6 \cdot 1024 \approx 15.32$ times higher than uploading speed to the fastest cloud, and data decoding

speed $24.5/2687.58 \cdot 1024 \approx 9.33$ times higher than the downloading speed.

TABLE III. THE AVERAGE SPEED DATA CODING AND DECODING (MB/S)

Data size	Coding			Decoding		
	Min	Average	Max	Min	Average	Max
10 KB	29.8	30.6	31.0	27.4	29.2	30.2
100 KB	30.2	30.4	30.5	27.3	27.8	29.1
1 MB	27.7	28.5	29.3	24.5	26.6	28.2
10 MB	26.3	27.1	27.6	26.7	27.8	30.1
100 MB	25.6	26.2	26.7	25.0	26.4	27.6

TABLE IV. THE AVERAGE UPLOAD VELOCITY V_u (KB/S)

Data size	Speed				Min increase %
	BU	V_u			
		Min	Average	Max	
10 KB	9.8	11.18	18.46	39.15	14.08
100 KB	81.0	104.8	194.58	320.67	29.38
1 MB	411.8	599.54	1013.66	1561.47	45.59
10 MB	1137.7	1412.98	2723.82	3919.66	24.20
100 MB	1711.6	2172.87	3967.61	5449.14	26.95

TABLE V. THE AVERAGE DOWNLOAD VELOCITY V_d (KB/S)

Data size	Speed				Min increase %
	BD	V_d			
		Min	Average	Max	
10 KB	10.36	10.77	19.86	41.38	3.96
100 KB	123.44	145.60	221.57	485.71	17.95
1 MB	1063.63	1371.78	1980.08	3708.18	28.97
10 MB	2123.94	2163.15	4148.22	6659.02	1.85
100 MB	2676.82	4923.12	7682.47	2676.82	83.92

The encryption and decryption times are neglected due to the high speed of XOR operation. To determine the practical applicability of the proposed scheme and study its properties, we study various scenarios. In BU and BD scenarios, only one storage with the fastest uploading and downloading speeds are used for experiments. Tables IV and V show data access speeds using the proposed scheme compared with BU and BD. We see that the average uploading and downloading velocities are increased by 28.04% (Table IV) and 27.33% (Table V), respectively.

VII. CONCLUSIONS

In this paper, we introduce a distributed data storage scheme based on weighted TL-SSS and RNS with moduli of a special type to improve reliability. We also reduce the computational complexity of the algorithms for encoding and decoding data from linear-logarithmic to linear compared with the one-level scheme. TL-SSS is the weighted data access structure. We propose to distribute data shares in such a way that more shares are allocated to the more reliable storages. We study various data access scenarios exploring the properties of TL-SSS and show that our scheme restores data in more cases than traditional one-level RNS. Partial loss of information, denial of access, technical failures, etc. in one or several storages do not lead to loss of data. In contrast to classical solutions, our TL-SSS can restore the data with less available shares than the state of the art approaches. However, further study of the algorithm and multicriteria analysis are required to assess their actual efficiency and effectiveness. This will be subject of future work for improving the technical characteristics of security, redundancy, and speeds in cloud storages. In real systems, characteristics of the cloud storages can be dynamically changed. The number of shares and their distribution can be adjusted to cope with the dynamic situation and different reliability requirements. To this end, the past technical characteristics and failures statistics of cloud providers must be analyzed for a certain time interval.

ACKNOWLEDGMENTS

The work is partially supported by Russian Foundation for Basic Research (RFBR) 18-07-01224, Russian Federation President Grants MK-341.2019.9, MK-6294.2018.9, and SP-2236.2018.5.

REFERENCES

[1] E. Deelman and A. Chervenak, "Data Management Challenges of Data-Intensive Scientific Workflows," in 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID), 2008, pp. 687–692.

[2] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The cost of doing science on the cloud: the montage example," in SC'08: Proceedings of the 2008 ACM/IEEE conference on Supercomputing, 2008, pp. 1–12.

[3] M. Babenko et al., "Unfairness correction in P2P grids based on residue number system of a special form," in 2017 28th International Workshop on Database and Expert Systems Applications, 2017, pp. 147–151.

[4] A. Szalay and J. Gray, "Science in an exponential world," Nature, vol. 440, no. 7083, pp. 413–414, Mar. 2006.

[5] N. Chervyakov, M. Babenko, A. Tchernykh, N. Kucherov, V. Miranda-López, and J. M. Cortés-Mendoza, "AR-RRNS: Configurable reliable distributed data storage systems for Internet of Things to ensure security," Futur. Gener. Comput. Syst., vol. 92, pp. 1080–1092, 2019.

[6] A. Tchernykh et al., "AC-RRNS: Anti-collusion secured data sharing scheme for cloud storage," Int. J. Approx. Reason., vol. 102, pp. 60–73, Nov. 2018.

[7] Wei Wang, M. N. S. Swamy, M. O. Ahmad, and Yuke Wang, "A high-speed residue-to-binary converter for three-moduli $(2^k, 2^k - 1, 2^{k-1} - 1)$ RNS and a scheme for its VLSI implementation," IEEE Trans. Circuits Syst. II Analog Digit. Signal Process., vol. 47, no. 12, pp. 1576–1581, 2000.

[8] A. R. Omondi and B. Premkumar, Residue number systems: theory and implementation, vol. 2. World Scientific, 2007.

[9] P. V. A. Mohan, "Residue Number Systems," in Algorithms and Architectures, Springer, 2002.

[10] A. Tchernykh, M. Babenko, V. Miranda-Lopez, A. Y. Drozdov, and A. Avetisyan, "WA-RRNS: Reliable Data Storage System Based on Multi-cloud," in 2018 IEEE International Parallel and Distributed Processing Symposium Workshops, 2018, pp. 666–673.

[11] A. Tchernykh et al., "Performance evaluation of secret sharing schemes with data recovery in secured and reliable heterogeneous multi-cloud storage," Cluster Comput., Jan. 2019.

[12] S. Ghemawat, H. Gobioff, S.-T. Leung, "The Google file system.," ACM SIGOPS Oper. Syst. Rev., pp. 29-43, 2003.

[13] A. Celesti, M. Fazio, M. Villari, and A. Puliafito, "Adding long-term availability, obfuscation, and encryption to multi-cloud storage systems," J. Netw. Comput. Appl., vol. 59, pp. 208–218, 2016.

[14] Hsiao-Ying Lin and W.-G. Tzeng, "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding," IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 6, pp. 995–1003, 2012.

[15] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," IEEE Trans. Inf. Theory, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[16] C. Gentry, "Computing arbitrary functions of encrypted data," Commun. ACM, vol. 53, no. 3, p. 97, 2010