Adaptive Encrypted Cloud Storage Model

Esteban Lopez-Falcon[1], Andrei Tchernykh[2], CICESE Research Center, Ensenada, Mexico, {[1]esteban, [2]chernykh}@cicese.edu.mx

Nikolay Chervyakov[3], Mikhail Babenko[4], Elena Nepretimova[5], North-Caucasus Federal University, Stavropol, Russia, {[3]ncherviakov, [4]mgbabenko}@ncfu.ru, [5]nev1973@mail.ru

Vanessa Miranda-López[6], CICESE Research Center, Ensenada, BC, México, [6]vanessamir.2813@gmail.com

Alexander Yu. Drozdov[7], Moscow Institute of Physics and Technology (State University), Russia, [7]alexander.y.drozdov@gmail.com

Gleb Radchenko[8], South Ural State University, Chelyabinsk, Russia, [8]gleb.radchenko@susu.tu

Arutyun Avetisyan[9], Institute for System Programming of the RAS, Moscow, Russia, [9]arut@ispras.ru

***Abstract*— In this paper, we propose an adaptive model of data storage in a heterogeneous distributed cloud environment. Our system utilizes the methods of secret sharing schemes and error correction codes based on Redundant Residue Number System (RRNS). We consider data uploading, storing and downloading. To minimize data access, we use data transfer mechanism between cloud providers. We provide theoretical analysis and experimental evaluation of our scheme with six real data storage providers. We show how dynamic adaptive strategies not only increase security, reliability, and reduction of data redundancy but allow processing encrypted data. We also discuss potentials of this approach, and address methods for mitigating the risks of confidentiality, integrity, and availability associated with the loss of information, denial of access for a long time, and information leakage.**

***Keywords*— cloud storage; Redundant Residue Number System; secret sharing schemes; security; reliability**

I. INTRODUCTION

Since the introduction of the cloud computing, people have doubted its security. It is not surprising that even cloud computing is well established, there is a significant amount of work regarding cloud security.

AlZain et al. [1] stated that ensuring the security of cloud computing is a major factor in the cloud computing environment, as users often store sensitive information. The lack of trust towards a single service provider is one of the issues that led to the popularity of multi-cloud environments.

Chervyakov et al. [2] presented a study in the field of cloud storage security, where a system based on RRNS with a (k, n) configuration is proposed. In the (k, n) RRNS, a file is divided into n pieces and with k chunks or more, data can be recovered.

In this work, we consider a set of real cloud providers whose upload and download speed are different and vary in time. The goal is to maximize the upload and download speed of the system.

The paper is structured as follows. Section II briefly reviews related works. Section III describes our model of distributed storage and presents uploading and downloading strategies. Section IV discusses the experimental results. Section V concludes the work.

II. RELATED WORK

Security and reliability of cloud computing are issues that have been widely studied. Ratham et al. [3] used an adaptive Huffman technique and an RSA double-encryption algorithm. Babita et al. [4] used the AES (Advanced Encryption Standard). Both encryption algorithms are well known and widely accepted. Moreover, there is a large number of works that adapt these encryption algorithms to specific implementations.

Another approach to improve the availability, confidentiality of the stored information is DEPSKY [6]. This system combines several techniques: encryption, coding, and replication of data in different clouds. By replicating the encrypted information and distributing the keys among the various storage providers, DEPSKY maintains a high level of confidentiality. However, the monetary cost of using DEPSKY is approximately twice as much as traditional systems.

In recent years, security work on cloud storage systems with RNS has grown in popularity [7-9].

Celesti et al. [10] combined RRNS with AES-256 to add an extra layer of security. A brief explanation of the system would be as follows. When a file is saved to the cloud storage, it is compressed, and then RRNS is used to divide the compressed file into different pieces. Then, each piece is coded to base-64. Depending on the number of pieces that are downloaded and the redundancy, the Chinese remainder theorem is used. Then, the decoding with AES reconstructs the file.

Chervyakov et al. [1] presented similar to Celesti et al. The main difference is that it uses a strategy called an approximation of the range of an RNS, which can reduce the complexity of O ($L \log L \log \log L$) to O (1), where L is the file size.

As an emerging technology, the multi-cloud has come with issues such as loss of information, denial of access for a long time, and information leakage [10, 14].

To tackle some of these problems, researchers have proposed the encryption, replication, and coding techniques. A particular result that plays an important role in our work is the algorithms proposed in [15, 16] that use RRNS with (k, n) settings.

The work mentioned above tackles a variety of challenges. However, adaptive security schemes that can change their strategies depending parameters of the multi-cloud environment are not addressed in the literature.

We propose a security system that uses the RRNS, where the (k, n) settings directly related with the probability of information loss, data redundancy, and speed of encoding-decoding, can be adapted to the changing parameters.

Furthermore, once (k, n) setting is chosen and the data is encrypted and stored into the multi-cloud, the system can keep the best data distribution for downloading within the system.

III. Model

A. Residue Number System and its properties

The Residue Number System is a widely known number theory paradigm. It is used to represent a big integer by using smaller integers to reduce the complexity of certain computational operations. It is based on the Chinese Remainder Theorem discovered by Sunzi Suanjing in the third century [11].

Ferruccio Barsi presented the general view of the system. Given a set of n pairwise prime positive integers $m_1\ m_2 \dots m_n$ called moduli, the nonnegative integers X in the range $[0, M)$, where $M = m_1 * m_2 * \dots * m_n$ are uniquely represented by the n-tuples $x_1\ x_2 \dots x_n$ of their residues modulo m_i ($x_i = |X|_{m_i}, i = 1,2, \dots, n$) [12].

The Redundant Residue Number System uses a set of $(n + r)$-tuples to represent an integer in the range $[0, M)$. It uses $m_1\ m_2 \dots m_n\ m_{n+1} \dots m_r$ modulis and $x_1, x_2, \dots, x_n, x_{n+1}, \dots, x_r$ digits. $m_{n+1} \dots m_r$ and $x_{n+1}, \dots, x_r$ are called the redundant moduli and redundant digits, respectively.

B. Cloud Storage Systems

There exist many cloud service providers that offer free storage space.

Dropbox is a file hosting service created by company Dropbox, Inc. in 2007. It offers cloud storage, file synchronization, personal cloud and client software. Dropbox can create a folder in a cloud synchronized with the user folder, regardless of on which device it was created. Its security features include two-step authentication. Files are encrypted by AES 256-bit encryption performed by Dropbox. Besides, Dropbox uses Secure Sockets Layer (SSL) for data uploading and downloading. A drawback of Dropbox is that the bandwidth and disk storage design create an observable side channel through which a single bit of data can be observed [18].

OneDrive is a file hosting service that allows users to synchronize files and access to them from a web browser. It is created by Microsoft in 2007 as Windows Live Folders and later as Windows Live SkyDrive. OneDrive offers 5 GB of free storage. OneDrive accounts are encrypted using SSL. OneDrive for Business also offers per-file encryption, which gives each file its encryption key. A hacker can only have access to one individual file instead of the whole documents. On the other hand, OneDrive automatically synchronizes Outlook attachments to the Cloud. However, it brings a problem to OneDrive security because user mistakes are one of the biggest obstacles to use it.

Google Drive is a file storage and synchronization service created by Google in 2012. Google Drive offers 15 GB of free storage space shared between files, messages and attachments in Gmail, and pictures and videos in Google Photos. Google Drive uses Syncdocs to encrypt the files with AES 256-bit key. Keyed-hash message authentication code (SHA1-HMAC) is used to authenticate and verify the message as genuine and uncorrupted. To increase security, random data is added to the message to protect against cyphertext attacks.

Box is an online file sharing and content management service for businesses founded in 2005. Box offers a free space of 10 GB. Box uses SSL encryption, password protection, and advanced administrative controls for businesses. Box also offers expiration dates for shared files. This prevents users being able to access a file shared past a certain date.

iDrive is a cloud storage service with an app that manages backup, restore, file-syncing, and file-sharing applications. With the 1 TB of the subscription, iDrive uses AES 256-bit encryption on transfer and storage, and security capabilities of SSL. Data is stored using an encryption key. One of the best attributes of iDrive is that it does not store the user private key on the servers.

C. Description

Let us consider a set of m clouds $C = \{c_1, c_2, \dots, c_m\}$. Each cloud $c_j = \{u_j, d_j\}$ is characterized by the speed of uploading u_j and speed of downloading d_j, for all $j = \{1, \dots, m\}$.

An RRNS with a (k, n) setting, where the data $D = \{1,2, \dots, n\}$ is divided into n chunks to be stored. Each chunk $i = \{s_i\}$ has size s_i for each $i = \{1, \dots, n\}$

Let us use the following notation:

D	Original size of the data;
D_E	Size of the encrypted data;
s_i	Size of the i-th chunk;
s_{Ei}	Size of the encrypted i-th chunk;
u_j	Upload speed of the j-th cloud;
d_j	Download speed of the j-th cloud;
T_{up}	Encrypted data upload time;
T_{dow}	Encrypted data download time;
V_s	Upload velocity;
V_{ex}	Download velocity.

Upload velocity (V_s) represents how fast the data is stored. It's calculated as the original data size D over the upload time

$$T_{up} = \sum_{i=1}^{n} \frac{s_{Ei}}{u_i}, V_s = \frac{D}{T_{up}}$$

Download velocity (V_{ex}), represents how fast the data needed to recover the file is downloaded. In the worst case, it is calculated as the original data size D over the download time

$$T_{dow} = \sum_{i=n-k+1}^{n} \frac{s_{Ei}}{d_i}, V_{ex} = \frac{D}{T_{dow}}$$

For the worst case, we assume that chunks are downloaded from the clouds sequentially. The downloading terminates when k correct chunks of data are obtained. There are two main scenarios. The best case occurs when first k chunks of data are required. The worst case occurs when the system has to read all chunks due to correct data are stored in k slowest clouds, indexed from $n-k+1$ to n.

D. Strategies

To show how the selection of the (k, n) setting and selection of cloud providers affect the speed-wise performance of the system, we developed three strategies.

The Best/Any (BA) strategy selects first n available clouds with fastest upload speed. Download speed is not taken into account.

The Any/Best (AB) strategy, selects first n available clouds with the best download speed. Upload speed is not taken into account.

The Best/Best (BB) strategy selects first n available clouds best for uploading and n best clouds for downloading. After storing, data are moved to the best clouds for downloading.

IV. EXPERIMENTAL ANALYSIS

For the experimentations, we use CentOS server with a 300 Mbps symmetric internet connection located in Ensenada, Baja California, Mexico, and data center North-Caucasus Federal University.

We divided experiments into two sections. The first part of the experiments consists in a statistical analysis of the upload and download speed of each cloud provider. The second part includes analysis of mathematical model presented in Section II and strategies described in Section III using obtained results.

A. Upload and download speed

For the time lapse of three days, we uploaded and downloaded a 200 MB media file every hour to each cloud. To accomplish this, we developed a small script using the Java programming language.

TABLE I. UPLOADING TIME AND SPEED

	Upload						
	Min	Max	Average	σ	Max MBps	Min MBps	Average MBps
Box	38.79	52.47	40.80	1.76	1.29	0.95	1.23
Google Drive	38.80	51.65	40.83	2.12	1.29	0.97	1.22
Salesforce	38.64	54.62	41.12	2.66	1.29	0.92	1.22
Egnyte	38.65	86.59	42.47	6.46	1.29	0.58	1.18
DropBox	41.81	55.34	44.40	2.30	1.20	0.90	1.13
One Drive	41.82	88.63	45.57	6.37	1.20	0.56	1.10
Sharefile	41.40	98.50	55.58	15.72	1.21	0.51	0.90

The script is based on the Kloudless API [19] to have an easier way of managing the storage clouds. To access Kloudless API, we use the Apache HttpComponents library [20], which allowed us to simplify the calls to the Kloudless API endpoints.

Table I shows the minimal, maximal and average upload time in seconds, standard deviation, and speeds of six clouds. Table II shows the same downloading information.

TABLE II. DOWNLOADING TIME AND SPEED

	Download						
	Min	Max	Average	σ	Max MBps	Min MBps	Average MBps
Box	38.73	58.58	41.34	3.79	1.29	0.85	1.21
Google Drive	38.48	95.34	42.33	8.42	1.30	0.52	1.18
One Drive	39.00	63.75	42.48	4.16	1.28	0.78	1.18
DropBox	38.79	86.16	42.77	7.00	1.29	0.58	1.17
Salesforce	39.05	92.17	42.96	8.49	1.28	0.54	1.16
Sharefile	39.76	86.89	43.53	6.08	1.26	0.58	1.15
Egnyte	38.43	85.42	48.66	11.93	1.30	0.59	1.03

Tables III-VIII show the maximum, minimum, and average speeds of data transfer between cloud providers obtained in our experiments.

TABLE III. DATA TRANSFER SPEED FROM BOX

	Box				
	Max	Min	Average	Average+σ	Average-σ
One Drive	2,28	0,82	1,82	1,44	2,48
Google Drive	3,12	1,12	2,37	1,80	3,48
ShareFile	2,40	0,61	1,47	1,02	2,65
Egnyte	3,05	1,45	2,44	1,99	3,15
Salesforce	1,71	0,66	1,53	1,22	2,05

TABLE IV. DATA TRANSFER SPEED FROM ONE DRIVE

	One Drive				
	Max	Min	Average	Average+σ	Average-σ
Box	3,01	1,12	2,09	1,66	1,76
Google Drive	3,68	1,63	2,62	2,14	2,29
ShareFile	2,32	0,50	1,39	0,89	1,51
Egnyte	3,28	1,97	2,78	2,46	1,10
Salesforce	1,71	1,03	1,48	1,31	1,07

TABLE V. DATA TRANSFER SPEED FROM GOOGLE DRIVE

	Google Drive				
	Max	Min	Average	Average+σ	Average-σ
Box	2,70	0,66	1,85	1,30	3,19
One Drive	2,95	0,45	2,03	1,17	7,58
ShareFile	2,43	0,63	1,50	0,97	3,28
Egnyte	3,81	2,19	3,15	2,75	3,67
Salesforce	1,61	0,55	1,36	0,98	2,25

TABLE VI. DATA TRANSFER SPEED FROM SHAREFILE

	ShareFile				
	Max	Min	Average	Average+σ	Average-σ
Box	1,56	0,25	0,96	0,57	3,09
One Drive	1,18	0,23	0,76	0,49	1,68
Google Drive	1,46	0,57	1,03	0,78	1,50
Egnyte	1,38	0,57	1,06	0,82	1,48
Salesforce	1,46	0,52	1,07	0,81	1,55

TABLE VII. DATA TRANSFER SPEED FROM EGNYTE

	Egnyte				
	Max	Min	Average	Average+σ	Average-σ
Box	3,19	2,13	2,13	2,94	2,37
One Drive	3,39	1,61	1,61	7,13	2,18
Google Drive	3,62	2,02	2,02	3,82	2,61
ShareFile	1,40	0,62	0,62	1,95	0,89
Salesforce	1,34	1,21	1,21	1,42	1,25

TABLE VIII. DATA TRANSFER SPEED FROM SALESFORCE

	Salesforce				
	Max	**Min**	**Average**	**Average+σ**	**Average-σ**
Box	1,39	1,21	1,32	1,28	1,36
One Drive	1,12	0,84	1,05	0,98	1,13
Google Drive	1,32	1,06	1,22	1,16	1,29
ShareFile	1,18	0,26	0,77	0,50	1,73
Egnyte	1,37	1,05	1,26	1,20	1,34

B. Analysis

To see the difference between strategies, we use the model described in Section III, and the average upload and download speeds.

We consider $n = 6$ and variation of $k = 2..6$, ignoring the case, where $k = 1$ as non-distributed case.

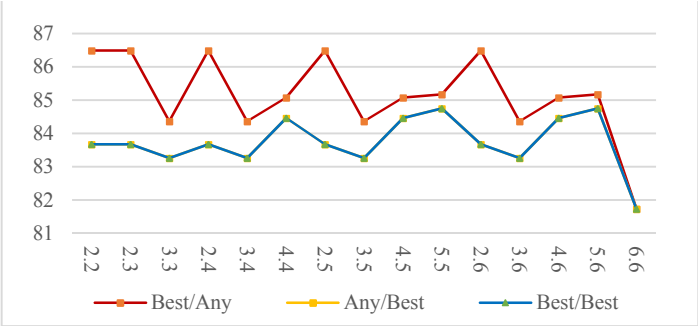

Figure 1. Download time of the strategies versus (k, n)

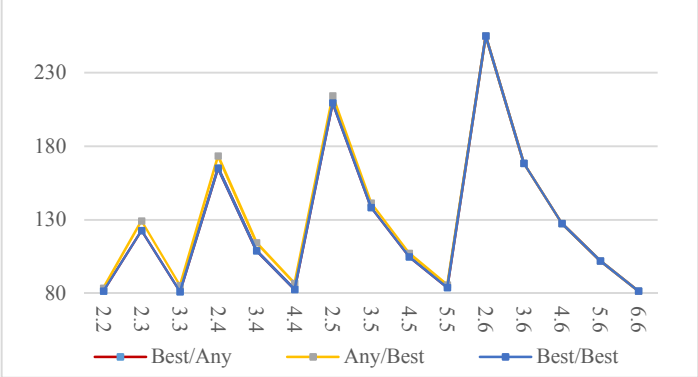

Figure 2. Upload time of the strategies versus (k, n)

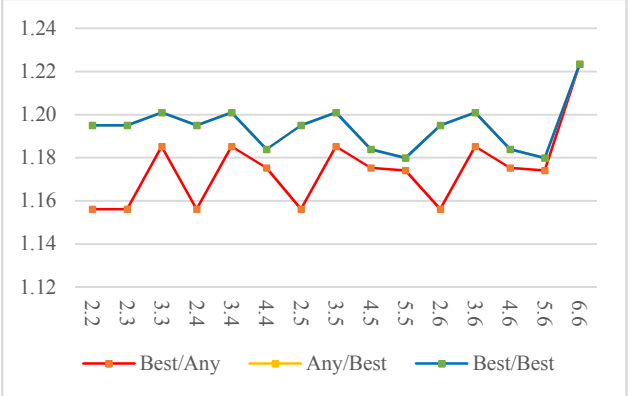

Figure 3. Download speed of the strategies versus (k, n)

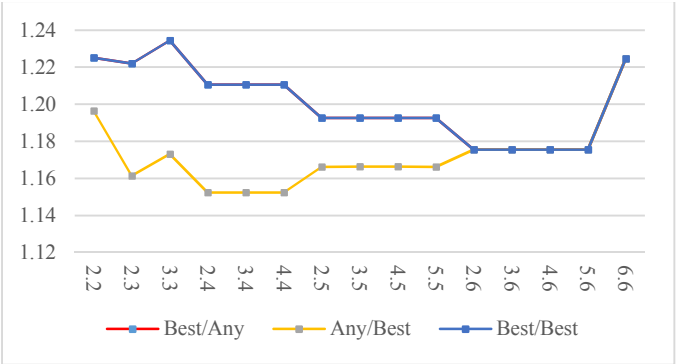

Figure 4. Upload speed of the strategies versus (k, n)

TABLE IX. UPLOAD AND DOWNLOAD TIME REDUCTION OF BB VS BA AND AB

(k,n)	**BB vs BA download time reduction %**	**BB vs AB upload time reduction %**
(2,2)	3.25	2.34
(2,3)	3.25	4.96
(3,3)	1.31	4.96
(2,4)	3.25	4.81
(3,4)	1.31	4.81
(4,4)	0.73	4.81
(2,5)	3.25	2.21
(3,5)	1.31	2.21
(4,5)	0.73	2.21
(5,5)	0.49	2.21
(2,6)	3.25	0.00
(3,6)	1.31	0.00
(4,6)	0.73	0.00
(5,6)	0.49	0.00
(6,6)	0.00	0.00

TABLE X. UPLOAD AND DOWNLOAD SPEED INCREASE OF BB VS BA AND AB

(k,n)	**BB vs BA download speed increase %**	**BB vs AB upload speed increase %**
(2,2)	3.36	2.39
(2,3)	3.36	5.22
(3,3)	1.33	5.22
(2,4)	3.36	5.05
(3,4)	1.33	5.05
(4,4)	0.73	5.05
(2,5)	3.36	2.26
(3,5)	1.33	2.26
(4,5)	0.73	2.26
(5,5)	0.50	2.26
(2,6)	3.36	0.00
(3,6)	1.33	0.00
(4,6)	0.73	0.00
(5,6)	0.50	0.00
(6,6)	0.00	0.00

From the results of the study, we can draw the following conclusions:

1. The highest speed of cloud access demonstrate (2,2), (3,3), (4,4), (5,5), (6,6). However, schemes of the form (n, n) have the highest probability of failure.

2. Tables III-VIII demonstrates high variability of the data transfer speeds between clouds from 3.61 to 0.23 MBs.

3. Since we restrict our self to six cloud providers, the strategies AB, BA, and BB show the same results on the schemes: (2,6), (3,6), (4,6), (5,6), (6,6).

V. Conclusions

We propose an adaptive model of data storage in a heterogeneous distributed cloud environment. For best load balancing among cloud providers, we use secret sharing schemes based on residue number system. The proposed approach allows data access time minimization due to dynamic data transfers between clouds.

Data transfer from one cloud provider to another increases the probability of data distortion and errors. To detect, localize and correct errors, we use error correction code mechanisms based on RNS.

Theoretical analysis and experimental evaluation of our scheme with six real data storage providers show speed improvement up to 3.36 percent comparing with known static approaches.

In the future work, we will study the potential of this approach and consider methods to reduce the risks of confidentiality, integrity and accessibility loss associated with loss of information, long-term denial of access and leakage.

To increase the speed of data access, we suggest to use the specialized API of each cloud provider. For the case of Google [21], Dropbox [22], Box [23] and Sharefile [24], we use their java wrapper for public REST API. Whereas for OneDrive [25], Egnyte [26], and Salesforce [27], we use the Apache HttpClient library [28] to access the public REST API.

Moreover, weighted secret sharing schemes can be used. However, it could impose restrictions of the parameters of the system.

Acknowledgment

The work is partially supported by Russian Federation President Grant SP-1215.2016, and Russian Foundation for Basic Research (RFBR) 18-07-01224.

References

[1] M. A. AlZain, E. Pardede, B. Soh, J. A. Thom, "Cloud computing security: from single to multi-clouds," in *IEEE 45th Hawaii International Conference on System Science (HICSS)*, 2012, pp. 5490-5499. DOI: 10.1109/HICSS.2012.153

[2] N. Chervyakov, M. Babenko, A Tchernykh, N. Kucherov, V. Miranda-López, J. M. Cortés-Mendoza, "AR-RRNS: Configurable reliable distributed data storage systems for Internet of Things to ensure security," *Future Generation Computer Systems*, 2017. DOI: 10.1016/j.future.2017.09.061

[3] C. Chang, A. Molahosseini, A. Zarandi, T. Tay, "Residue Number Systems: A New Paradigm to Datapath Optimization for Low-Power and High-Performance Digital Signal Processing Applications," *IEEE Circuits and Systems Magazine*, vol. 15, no. 4, pp. 26-44, 2015. DOI: 10.1109/MCAS.2015.2484118

[4] G.J. Rathanam, M.R. Sumalatha, "Dynamic secure storage system in cloud services," in Recent Trends in IEEE 2014 International Conference on Information Technology (ICRTIT), 2014, pp. 1-5. DOI: 10.1109/ICRTIT.2014.6996175

[5] M.P. Babitha, K.R.R. Babu, "Secure cloud storage using AES encryption," in IEEE International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), 2016, pp. 859-864. DOI: 10.1109/ICACDOT.2016.7877709

[6] A. Bessani, M. Correia, B. Quaresma, P. Sousa, F. André, P. Sousa, "DepSky: dependable and secure storage in a cloud-of-clouds," *ACM Transactions on Storage (TOS)*, vol. 9, no. 4, p. 12, 2013. DOI: 10.1145/2535929

[7] N. I. Chervyakov, M. G. Babenko, N. N. Kucherov, A. I. Garianina, "The effective neural network implementation of the secret sharing scheme with the use of matrix projections on FPGA," *Lecture Notes in Computer Science*, vol. 9142, pp. 3-10, 2015. DOI: 10.1007/978-3-319-20469-7_1

[8] V. Miranda-López, A. Tchernykh, J. M. Cortés-Mendoza, M. Babenko, G. Radchenko, S. Nesmachnow, Z. Du, "Experimental Analysis of Secret Sharing Schemes for Cloud Storage Based on RNS," *Communications in Computer and Information Science*, vol 796, pp. 370-383, 2017. DOI: 10.1007/978-3-319-73353-1_26

[9] A. Tchernykh, U. Schwiegelsohn, E. G. Talbi, M. Babenko, "Towards understanding uncertainty in cloud computing with risks of confidentiality, integrity, and availability," *Journal of Computational Science*, 2016. DOI: 10.1016/j.jocs.2016.11.011

[10] A. Celesti, M. Fazio, M. Villari, A. Puliafito, "Adding long-term availability, obfuscation, and encryption to multi-cloud storage systems. *Journal of Network and Computer Applications*, vol. 59, pp. 208-218, 2016. DOI: 10.1016/j.jnca.2014.09.021

[11] N. S. Szabo, R. I. Tanaka, "Residue arithmetic and its applications to computer technology," McGraw-Hill, 1967.

[12] F. Barsi, P. Maestrini, "Error correcting properties of redundant residue number systems," *IEEE Transactions on Computers*, vol. 100, no. 3, pp. 307-315, 1973. DOI: 10.1109/T-C.1973.223711

[13] A. Tchernykh, M. Babenko, N. Chervyakov, J. M. Cortés-Mendoza, N. Kucherov, V. Miranda-López, M. Deryabin, I. Dvoryaninova, G. Radchenko, "Towards mitigating uncertainty of data security breaches and collusion in cloud computing," in IEEE *28th International Workshop on Database and Expert Systems Applications (DEXA),* 2017, pp. 137-141. DOI: 10.1109/DEXA.2017.44

[14] N. Chervyakov, M. Babenko, A. Tchenykh, I. Dvoryaninova, N. Kucherov, "Towards reliable low cost distributed storage in multi-clouds," in IEEE *2017 International Siberian Conference on Control and Communications (SIBCON),* 2017, pp. 1-6. DOI: 10.1109/SIBCON.2017.7998476

[15] M. Babenko, N. Chervyakov, A. Tchernykh, N. Kucherov, M. Shabalina, I. Vashchenko, G. Radchenko, D. Murga, "Unfairness correction in P2P grids based on residue number system of a special form," in *IEEE 28th International Workshop on Database and Expert Systems Applications (DEXA), 2017,* pp. 147-151. DOI: 10.1109/DEXA.2017.46

[16] M. Babenko, N. Kucherov, A. Tchernykh, N. Chervyakov, , E. Nepretimova, I. Vashchenko, "Development of a Control System for Computations in BOINC with Homomorphic Encryption in Residue Number System," in *Proceedings of the Third International Conference BOINC:FAST 2017,* pp. 77-84.

[17] M. Casserly, 2016, "14 Best Cloud Storage Services 2016: Dropbox vs Google Drive - Test Centre - PC Advisor." [Online]. Available: http://www.pcadvisor.co.uk/test-centre/internet/14-best- cloud-storage-services-2016-uk-copy-3614269/. [Accessed: 01-Jan-2018].

[18] I. Drago, M. Mellia, M. Munafo, A. Sperotto, R. Sadre, A.Pras, "Inside dropbox: understanding personal cloud storage services," in *Proceedings of the 2012 ACM conference on Internet measurement conference*, 2012, pp. 481-494. DOI: 10.1145/2398776.2398827

[19] "Kloudless | The Universal API Platform." [Online]. Available: https://kloudless.com/. [Accessed: 01-Jan-2018].

[20] "Apache HttpComponents." [Online]. Available: https://hc.apache.org/. [Accessed: 01-Jan-2018].

[21] "API Reference, Drive REST API v2, Google Developers." [Online]. Available: https://developers.google.com/drive/v2/reference/. [Accessed: 01-Jan-2018].

[22] “Dropbox for HTTP Developers.” [Online]. Available: https://www.dropbox.com/developers/documentation/http/overview. [Accessed: 01-Jan-2018].
[23] “Box Developers | Cloud Content Management APIs.” [Online]. Available: https://developer.box.com. [Accessed: 01-Jan-2018].
[24] “ShareFile API Documentation.” [Online]. Available: https://api.sharefile.com/rest/. [Accessed: 01-Jan-2018].
[25] “REST APIs.” [Online]. Available: https://docs.microsoft.com/en-us/onedrive/developer/rest-api/. [Accessed: 01-Jan-2018].
[26] “API Docs - Egnyte for Developers.” [Online]. Available: https://developers.egnyte.com/docs. [Accessed: 01-Jan-2018].
[27] “Salesforce APIs - developer.force.com.” [Online]. Available: https://developer.salesforce.com/page/Salesforce_APIs. [Accessed: 01-Jan-2018].
[28] L. Richardson, S. Ruby, “*RESTful web services*. O'Reilly Media, Inc." [Online]. Available: http://virtualpanic.com/anonymousftplistings/ebooks/ICT-COLLECTION/WEB/O'Reilly%20RESTful%20Web%20Services.pdf. [Accessed: 01-Jan-2018].