GLObfus: An Enhanced Data Security Method to Protect Numerical Data in Public Cloud Storage

D. I. George Amalarethinam and Lalu P. George

***Abstract*—Cloud is an attractive computing paradigm for providing different kinds of computing services. Without realizing all day long, people are using cloud computing and their components. In the same way, without user's knowledge the data can be stolen at any time. Every second huge amount of data is uploaded in the cloud worldwide. Data storage and retrieval is a tedious task, but cloud makes it easy for all levels of users and also provides many service advantages to the end users. Apart from all these benefits, cloud pulls back because of security of data uploaded in the cloud. Security of data is a vital research area in cloud, and needs to be addressed. To enhance the security of data in the cloud, this paper proposes an enhanced data obfuscation method-GLObfus to protect the numerical data which are uploaded to the cloud. The proposed obfuscation method uses different mathematical calculations and functions to convert the data into meaningless data. The numerical data prepared to add are obfuscated prior to storage in the cloud area. The proposed technique is applied as a cloud utility and hosted on a cloud platform as a service and examined with current obfuscation strategies with respect to time and security. From the effects, it's far discovered that the proposed obfuscation technique-GLObfus is more efficient than the current obfuscation strategies with respect to obfuscation time and de-obfuscation time and security level.**

***Index Terms*—Cloud security, cloud storage, data obfuscation, data security.**

I. Introduction

The word Cloud in computer technology is the virtualization of server and data taking place from the user's side. Through this cloud computing technology many existing systems which migrated to the cloud area could reduce their cost and improve work efficiency. Cloud characteristics like rapid elasticity, location independent resource pooling, self-service on demand base, etc. are making the user's desire more powerful [1]. Looking at the developments of internet in the last few years: the internet began in 80's and 90's, from 2012 to 2014 there were fast developments in the global internet. Between 2014 and 2016 it emerged with software as a service, where in the last three years 2016-2019, many things integrated with the internet of things (IoT) and robotics. Now, most of the researchers are behind Fog computing. Here, currently day by day new development is coming in the cloud area [2], [3].

The main approach in cloud computing is easy accessibility of data, but here it is assumed only allowed person is accessing the contents. So, only services from trusted cloud service providers are considered, but they can also steal the contents from their server. Security, therefore needs to be increased in every case [4], [5] and has become the main concern in network and cloud area [6].

The report of Verizon, 2018 Data Breach Investigations uncovered 53,000 security incidents and 2,216 data breach incidents [7]. In the cloud computing world, visibility and demand for service are closely linked. Day by day, new loopholes and attacks are coming in cloud infrastructure. Data needs to be secure in every aspect. Data security and data protection are different. In data security private information has to be safe. In the case of data protection, data needs to be safe from system failure or incidents like natural disaster and deletion of data. In all these cases, data needs to be safe [8]. Mostly encryption-decryption mechanism is taking place for data protection. Here, the intruder can hack the flow of the particular algorithm and key. Obfuscation is the method mainly used in the software field. Obfuscation alters the methods of techniques and variables. Similarly, one can use obfuscation techniques in data hiding and storing in unauthorized areas [9].

Here, another type of Obfuscation method is implemented in this proposed research. The original sample text goes through one style of interchange within the position of the inputs, thereafter applying the squares of every individual character, different types of keys are put in between them [10], [11]. Finally, some addition, subtraction and ASCII values result in a meaningless textual content. This obfuscation technique is typically used for numeric values like economic records, students mark sheets, number-based OTP, and many others. This obfuscation method is exceptionally secured as compared to different strategies within the identical degree. Two keys are utilized in special places. Those identical keys are utilized in de-obfuscation approach also. On both sides same keys are applied. So, this is considered as symmetric in nature. Two types of keys are used in this method. One key is taken from any secured key algorithm and other key is collected from each character's coefficient value. This coefficient value is from same method itself. So, intruders or hackers can't identify the real procedure of the method. Hence, this method is considered to be highly secured as compared to other types.

II. Related Works

Alessio Viticchi *et al*. [12] Obfuscation technique is normally used in software protection for hiding the software development codes. Dividing it into two major parts - code obfuscation and data obfuscation. This paper assesses the efficiency of VarMerge facts of obfuscation approach through

Manuscript received April 8, 2020; revised June 29, 2020.

D. I. George Amalarethinam and Lalu P. George are with Jamal Mohamed College, Bharathidasan University, Tamil Nadu, India (e-mail: di_george@ymail.com, icelalu@gmail.com).

DOI: 10.7763/IJCTE.2020.V12.1274

comparing the time to mount attack responsibilities on clean and obfuscated versions of applications written in C, and assessing the fulfilment charge inside the execution of the mission. The VarMerge approach has been selected amongst a fixed number of candidate strategies as one of the most effective ones and is relevant to C source code. The test involved 15 students from the Master Degree program in Computer engineering at Politecnico di Torino. The test revealed no significant difference in terms of attack success price among obfuscated and clear programs. This is especially because of the size of the pattern. The outcomes display that the presence of the VarMerge obfuscation is able to lessen by using six instances that assault efficiency.

Vaclav Kaczmarczyk *et al*. [13] describes that data stored in the database within the system can be protected. This is applied as a part of service provider utility for raised database control. As the database may be shared by means of numerous firms, it was essential to defend sensitive statistics against abuse by means of the service provider. Here, they have explained key generation, validation and distribution. The approach is primarily based on the server-side to take a look at the key validation hash dispatched within http request and guarantees the correctness essential to preserve records consistency. The validation hash value is then generated on the client aspect for each request and send to the server. Received hash is compared with the value saved through the server aspect and best if each value is equal, the server keeps with the net web page processing. On the contrary the request is refused. The defined process is carried out best for pages with sensitive records and therefore users without the key can get admission to non-sensitive statistics.

Hwangnam Kim *et al*. [14] proposed application of multiple symmetric cipher text key used alternatively in a distinct order. It makes use of lightweight cryptography method and reduces encryption-decryption overhead against a heavy single cryptographic approach and avoids comprehensive key extraction attacks. This applies to different scenarios as the sequences vary with time, cost and security. This design includes patterned cipher block (PCB), integrity verification method to find out forged cipher text, pattern information of handshaking protocol, two round communication keys, thereby making the most of pattern to deliver better cryptography.

III. Problem Definition and Motivation

Cloud computing provides everywhere access through the network. At the same time, anyone can access not only the public data, but also the private data. Here the major problem that arises is cloud security [15], [16]. Nowadays, even those who have a little knowledge in computer are also thinking about the safety of their data. Every day through media hundreds of hacking and data theft cases are coming up. In a recent news, Israel's NSO Group in a single WhatsApp call can easily hack phone data. At present there are more than 1.5 billion watts up users in this world. This means Android, IOS, Windows, Linux, Blackberry, etc. [17] are under the threat of internet security. So, tackling the security threats is major a task in the cloud area. The proposed work has been developed a powerful obfuscation mechanism for the cloud area [18], [19]. Each character undergoes different types of changes and applies different types of key**s** to get a powerful obfuscated data. This makes it difficult to find out the real data.

Despite the fact that cloud service carriers implement the satisfactory security requirements and enterprise certifications, storing records and crucial files on outside carrier vendors continually opens up risks. Any dialogue concerning statistics has to cope with security and privacy especially in relation to handling more sensitive data. The events in the code area and server space that led to statistics deletion and eventual shut down of the company should not be neglected [20], [21]. So, dependence on service providers carries potential risk of data leakage and security breach. To solve this kind of issues the implemented GLObfus method obfuscates the data prior to its entry in the cloud area of the service provider. So, user need not worry about data theft. Here the main aim is to maximize security of data within minimum time period.

IV. Public Cloud Data Security GLObfus Method

GLObfus mechanism is mainly used for public cloud area protection. Here the data are passing through the developed obfuscation technique and after undergoing different types of interchange, an obfuscated data is received. The real numerical data are taken from user's information. Then the size of the original data has been found. The value needs to be interchanged in odd and even position. Each position is taken and subtracted from its original value. The square is calculated and key is applied. The applied key value is divided and mod is taken to get the ASCII and cipher text.

A. LGO-Pseudo Code

1. Start
2. OT← Users' Original text
3. OS← length (OT)
4. for i←0 to OS
5. PST[i]←OT[i+1]
6. PST[i+1]←OT[i]
7. i=i+1
8. end for
9. for k←0 to OS
10. SBT[k]←PST[k]-k
11. SQR[k]←pow (SBT[k],2)
12. SQU[k] ← SQU[k]+K_i
13. DVD[k]← SQU[k]/256
14. MDL[k]← SQU[k]%256
15. OT[k]←asciichar(DVD[k])
16. end for
17. Cipher text← OT
18. End

B. Procedure of GLObfus Method

1) Sample mark sheet of the student

Users sample mark sheet is sent to the server. This data is the secret data of each user. Data needs to be stored safely from unauthorized access. Table I describes the sample data of the student mark sheet.

TABLE I: MARK SHEET

Si No	Sub 1	Sub 2	Sub 3	Sub 4	Sub 5
1	77	51	60	94	80
2	51	72	70	82	59
3	96	51	76	64	94
4	89	83	94	74	63

2) Interchange the values in odd position to even and vice versa

In the second procedure the odd and even position of the above sample data value is interchanged. The data in the odd position occupies the even position and vice versa. The position of each value is subtracted from the corresponding value. Through this interchange and subtraction, the intruder or hackers finds it difficult to identify the real data. Table II represents the interchanged subtracted value.

TABLE II: SUBTRACTED VALUE

Si No	Sub 1	Sub 2
1	SBT[0]← 51	SBT[1]←76
2	SBT[2]← 92	SBT[3]←57
3	SBT[4]← 47	SBT[5]←75
4	SBT[6]← 64	SBT[7]←65
5	SBT[8]← 51	SBT[9]←73
6	SBT[10]←41	SBT[11]←85
7	SBT[12]←52	SBT[13]←63
8	SBT[14]←75	SBT[15]←79
9	SBT[16]←78	SBT[17]←66
10	SBT[18]←45	SBT[19]←55

3) Find square (SQU) of each value in the MUL(i)

Here, the square of interchanged subtracted value is calculated. Now the secret key 432 is applied. The secret key is taken from any outside resources like highly encrypted key generating algorithm. Table III represents the squared value.

TABLE III: SQUARED VALUE

Si No	Sub 1	Sub 2
1	SQU[0]← 2601	SQU[1]← 5776
2	SQU[2]← 8464	SQU[3]← 3249
3	SQU[4]← 2209	SQU[5]← 5625
4	SQU[6]← 4096	SQU[7]← 4225
5	SQU[8]← 2601	SQU[9]← 5329
6	SQU[10]← 1681	SQU[11]← 7225
7	SQU[12]← 2704	SQU[13]← 3969
8	SQU[14]← 5625	SQU[15]← 6241
9	SQU[16]← 6084	SQU[17]← 4356
10	SQU[18]← 2025	SQU[19]← 3025

4) Key application, mod value and coefficient value

TABLE IV: MOD VALUE

Si No	Sub 1	Sub 2
1	MDL[0]←217	MDL[1]←64
2	MDL[2]←192	MDL[3]←97
3	MDL[4]←81	MDL[5]←169
4	MDL[6]←176	MDL[7]←49
5	MDL[8]←217	MDL[9]←129
6	MDL[10]←65	MDL[11]←233
7	MDL[12]←164	MDL[13]←49
8	MDL[14]←169	MDL[15]←17
9	MDL[16]←114	MDL[17]←180
10	MDL[18]←153	MDL[19]←129

Any key needs to be taken from outside key service algorithm. This key is reserved for the obfuscation process. Data obfuscation is happening through the application of this key. Here sample key 432 is applied on squared value. The mode of each value is found and divided by 256.This coefficient value is kept as a secret key. Both the keys are passed in the decryption process. Below Table IV shows the mod of each value divided by 256 and Table V displays the coefficient value secret key.

TABLE V: SECRET KEY

Si No	Sub 1	Sub 2
1	DVD[0]←11	DVD[1]←24
2	DVD[2]←34	DVD[3]←14
3	DVD[4]←10	DVD[5]←23
4	DVD[6]←17	DVD[7]←18
5	DVD[8]←11	DVD[9]←22
6	DVD[10]←08	DVD[11]←29
7	DVD[12]←12	DVD[13]←17
8	DVD[14]←23	DVD[15]←26
9	DVD[16]←25	DVD[17]←18
10	DVD[18]←09	DVD[19]←13

5) Obfuscated value

This Mod value is converted in to ASCII character code. Table VI represents this obfuscated ASCII code. This is the final version of obfuscated text. This obfuscated code can't be identified by any of the intruders. The obfuscated text carries symbols, alphabets, digits and prolonged ASCII codes. Numerical values within the plaintext are transformed into single code ASCII values. GLObfus produces distinctive ciphertext for same plain text that arrives more than once in the plain text.

TABLE VI: OBFUSCATED VALUE

Si No	Sub 1	Sub 2	Sub 3	Sub 4	Sub 5
1	┘	@	└	A	Q
2	⌐	▒	1	┘	Ü
3	A	Θ	ñ	1	⌐
4	◄	R	┤	Ö	Ü

C. De-obfuscation Method of GLObfus

De-obfuscation is the reverse procedure of obfuscation. To de-obfuscate is to convert an unintelligible program into an understandable form. Here, the obfuscated data are taken from the public cloud. The same key Ki is also taken from the public cloud. Considering the obfuscated text and the same procedures, each step is now reversed in nature. The obfuscated text is converted to ASCII values. Then secret key is applied from the public server. Squares, mod values, subtraction and positions interchange are in different steps. Finally, it results in the original text.

D. Performance Comparison by Obfuscation Time

TABLE VII: PERFORMANCE COMPARISON BY OBFUSCATION TIME (MILLISECONDS)

Size (MB)	BaseX	Base64	Hexadecimal	GLObfus
1	42	58	72	19
2	73	79	103	48
3	119	134	149	77
4	135	158	177	112
5	181	207	222	143
10	337	376	397	313
15	502	516	568	472

Table VII and Fig. 1 gives the overall performance contrast of obfuscation with existing obfuscation techniques together with BaseX, Base64 and Hexadecimal encoding [22], [23]. The time taken by means of the existing and proposed obfuscation technique is calculated. The end result suggests

that the proposed method has taken shorter time length than current methods for obfuscating different sizes of plain text.

The graph in Fig. 1 depicts that all these case sizes are directly proportional to time, the time is increased with proportional to the size This is normal in most of the processes. But,if it is compared with existing standard techniques the time taken for particular size is very less in GLObfus method. So, this method gives better performance.

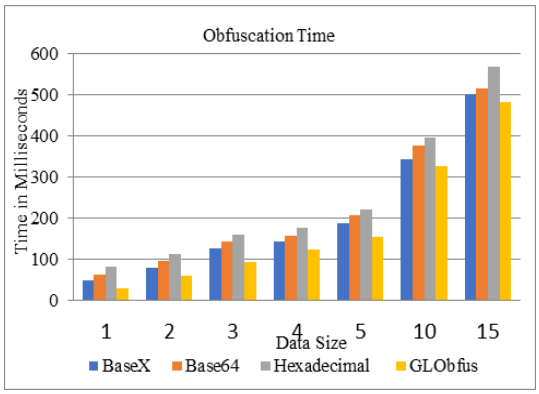

Fig. 1. Performance comparison by obfuscation time.

Table VIII and Fig. 2 represent the performance comparison of de-obfuscation with existing techniques based on the time. The time taken by the existing and proposed de-obfuscation technique is calculated for different sizes of plaintext. The result shows that the proposed technique has taken lesser time duration than existing technique for de-obfuscation.

TABLE VIII: PERFORMANCE COMPARISON BY DE-OBFUSCATION TIME (MILLISECONDS)

Size (MB)	BaseX	Base64	Hexadecimal	GLObfus
1	30	89	88	19
2	61	165	121	31
3	75	228	182	45
4	89	306	243	49
5	119	384	319	59
10	243	690	584	111
15	398	1033	868	124

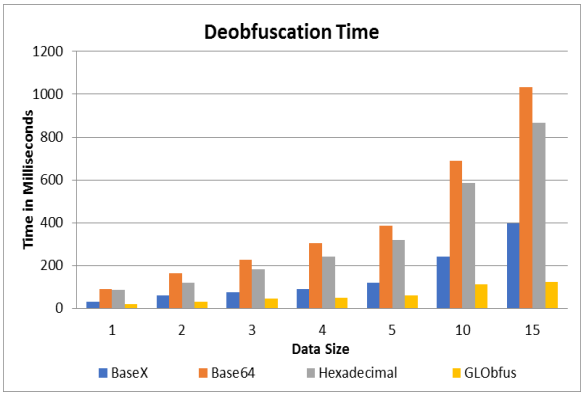

Fig. 2. Performance comparison by de-obfuscation time.

E. Comparison of GLObfus Techniques with Respect to Security

Security level is analyzed by using the ABC Hackman tool. This tool analyses the security level of proposed and existing obfuscation techniques. Performance and security level of proposed GLObfus method are compared with existing obfuscation techniques. Table IX and Fig. 3 constitute the contrast of safety level with existing techniques. The consequences display that the security degree of GLObfus is 93% and safety level of current obfuscation techniques is 69%, 81% and 71% for BaseX, Base64 and Hexadecimal Encoding respectively.

Here, GLObfus has the maximum-security level when compared with the other base obfuscation methods. So, among the confidential numerical data hiding in the public cloud storage, GLObfus proves to be the best method.

TABLE IX: COMPARISON OF SECURITY LEVEL OF EXISTING AND PROPOSED

Si. No.	Obfuscation Technique(s)	Security Level (%)
1.	BaseX	71
2.	Base64	81
3.	Hexadecimal Encoding	69
4.	GLObfus	93

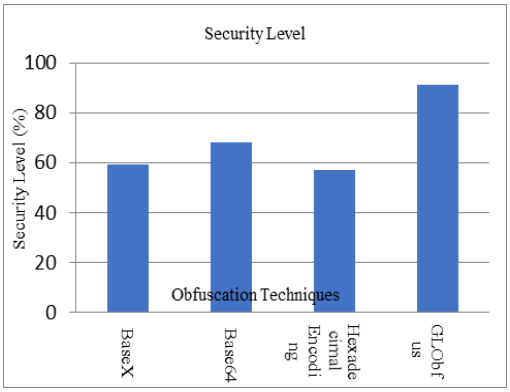

Fig. 3. Performance comparison by security level.

V. SIMULATION AND RESULT

The proposed GLObfus method is carried out in C#. Net. Simulation is performed within the cloud environment home windows azure cloud platform. The performance of the proposed GLObfus is calculated from the point taken for obfuscation and de-obfuscation in the users' system. The records are submitted to the GLObfus, after which they are obfuscated and uploaded to the cloud storage server. Safety ranges of the proposed and current obfuscation strategies which include Base64, BaseX and Hexadecimal encoding are computed on cloud servers.

The plaintext only contains numerical values, it is obfuscated and result is shown in Fig. 4.

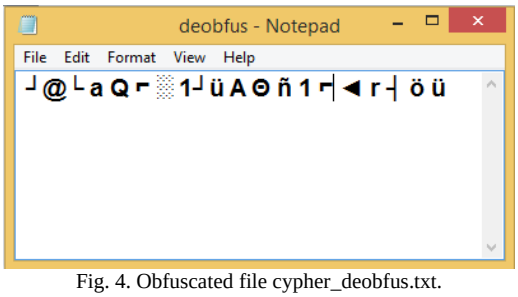

Fig. 4. Obfuscated file cypher_deobfus.txt.

GLObfus method reduces the dimensions of original information after obfuscation. For simulation, GLObfus set of rules gets plain text in a text record, for example, obfus.txt. GLObfus isn't only enhancing the security of the facts, however it additionally reduces the whole size of the plaintext. Fig. 5 indicates the plaintext statistics in a textual content report.

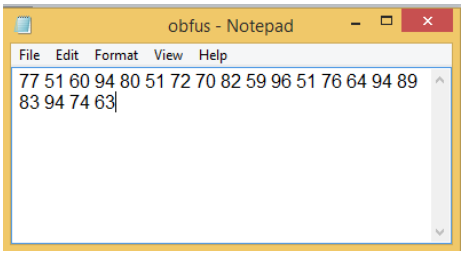

Fig. 5. Original data file Obfus.txt.

VI. Conclusion

In obfuscation, the real data appears unclear or unintelligible. GL-Obfuscation technique is one of the best methods as compared to other data obfuscation methods. Numerical data can go through different types of subtraction, division and application of keys and finally getting an obfuscated text. Same keys are used in obfuscation and de-obfuscation method. So, this method is symmetric in nature. Hackers or intruders find it very difficult to identify the real text. Here security level has been increased highly. Obfuscation time is reduced as compared to other technologies. So, the overall size of the data is also decreased. In future, the same method can be used for audio and video files also. The overall execution performance is very high in nature. So, this method is recommended as one of the best obfuscation methods when compared to other standard services.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Dr. D. I. George Amalarethinam revealed the significance of this studies and also insisted to refer the associated papers. Lalu P. George analyzed the requirements, developed it, then performed the studies and wrote the documentation. Dr. D.I. George Amalarethinam seriously reviewed and guided the content material of this article. Both the authors had accepted the final version.

References

[1] S. Tonyali, O. Cakmak, and K. Akkaya, "Secure data obfuscation scheme to enable privacy-preserving state estimation in smart grid ami networks," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp-709-719, October 2016.
[2] L. P. George and D. I. G. Amalarethinam, "Challenges in cloud computing security," *Book code AERA*, vol 2, pp. 80-90, 2017.
[3] J. H. Adkoli. (2018). [Online]. Available: https://dzone.com/articles/cloud-security-why-is-it-more-important-than-ever
[4] C. T. Boabaid. (2017). [Online]. Available: https://mse238blog.stanford.edu/2017/07/cboabaid/cloud-computing-security-and-new-ideas-to-make-us-feel-safer/
[5] S. Monikandan and L. Arockiam, *Efficient Cloud Storage Confidentiality to Ensure Data Security*, 2014.
[6] Y. Zhang, C. Xu, X. Lin, and X. Shen, "Blockchain-based public integrity Verification for cloud storage against procrastinating auditors," *IEEE Transactions on Cloud Computing*, pp. 1-15, March 2019.
[7] AndrewGriffin. (2019). [Online]. Available: https://www.independent.co.uk/life-style/gadgets-and-tech/news/whats/app-update-how-to-iphone-android-bug-hack-attack-security-a8912591.html
[8] W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for ecure cloud storage," *IEEE Transactions*, vol. 14, no. 2, pp-331-346, February 2019.
[9] D. E. Bakken, R. Parameswaran, D. M. Blough, A. A. Franz, and T. J. Palmer, "Data obfuscation: Anonymity and desensitization of usable data sets," *IEEE Security & Privacy*, vol. 2, issue 6, pp. 34-41, Dec. 2004.
[10] C. K. Behera and D. L. Bhaskari, "Different obfuscation techniques for code protection," *Procedia Computer Science*, vol. 70, pp. 757–763 Nov 2015.
[11] C. Hu, P. Liu, R. Yang, and Y. Xu, *Public-Key Encryption with Keyword Search via Obfuscation*, March 2019.
[12] A. Viticchi, L. Regano, M. Torchiano, C. Basile, M. Ceccato, P. Tonella, and R. Tiella, "Assessment of source code obfuscation techniques," *IEEE Computer Society*, pp. 11-20, December 2016.
[13] V. Kaczmarczyk, Z. B. P. Fiedler, and J. Arm, "Client side data encryption/decryption for web application," *IFAC-PapersOnLine*, vol. 49, issue 25, pp. 241-246, October 2016.
[14] S. Oh, S. Park, and H. Kim, "Patterned cipher block for low-latency secure communication," *IEEE Access*, vol. 8, pp. 44632-44642, March 2020.
[15] Z. Guan, G. Si, J. Wu, L. Zhu, Z. Zhang, and Y. Ma, "Utility-privacy tradeoff based on random data obfuscation in internet of energy," *IEEE Access*, vol. 5, pp. 3250-3262, February 2017.
[16] S. Koteshwara and C. H. Kim, and K. K. Parhi, "Key-based dynamic functional obfuscation of integrated circuits using sequentially triggered mode-based design," *IEEE Transactions*, vol. 13, issue 1, pp. 79-73, January 2018.
[17] PedroHernandez. (2018). [Online]. Available: https://www.enterprisestorageforum.com/storage-management/data-storage-security-guide.html
[18] L. P. George, D. I. G. Amalarethinam, A. S. Chandran, *GLEnc Algorithm to Secure Data in Public Cloud Environment*, 2018.
[19] G. Mogos, "Ciphertext-policy attribute-based encryption using quantum multilevel secret sharing scheme," *IAENG International Journal of Computer Science*, vol. 45, no. 4, pp. 500-504, 2018.
[20] L. Zhou and C. Li, "Outsourcing large-scale quadratic programming to a public cloud," *IEEE Access*, vol. 3, pp. 2581-2589, 2015.
[21] H. Zhao, Z. Chang, W. Wang, and X. Zeng, "Malicious domain names detection algorithm based on lexical analysis and feature quantification," *IEEE Access*, vol. 7, pp. 128990-128999, 2019.
[22] S. Wen and W. Dang, *Research on Base64 Encoding Algorithm and PHP Implementation*, December 2018.
[23] X. Li, S. Tang, L. Xu, H. Wang, and J. Chen, "Two-factor data access control with efficient revocation for multi-authority cloud storage systems," *IEEE Access*, vol. 5, pp. 393-405, 2017.

D. I. George Amalarethinam is currently a bursar and the director of MCA Program and an associate professor at the Department of Computer Science, Jamal Mohamed College, Tiiruchirappalli, Tamil Nadu, India. His areas of interests are parallel processing, distributed databases, grid computing, cloud computing and big data analytics. He has published more than 100 articles in the reputed journals and also a reviewer of reputed journals. He chaired many conferences and authored some books. He has acted as resource person for various national and international conferences. He is also leading many academic committees. He has produced nearly fifteen PhD scholars and guiding eight.

Lalu P. George is working as a project manager in Dects Technologies LLP, Cochin, Kerala, India. He has more than 15 years of industry experience in different software development and software management field. He worked in different countries and handled different types of projects in the software area. He was a visiting lecturer of different colleges in India. He is currently pursuing his Phd degree with Bharathidasan University as a research scholar. His areas of interests are cloud security, cloud computing, analysis of algorithms. He has published nearly five papers in different international journals, including IEEE and Scopus indexed. He has attended some international and national conference.