Rounds Reduction and Blocks Controlling to Enhance the Performance of Standard Method of Data Cryptography

Mua'ad M.Abu-Faraj[1†] and Ziad A. Alqadi[2††]

The University of Jordan Albalaqa Applied University

Summary
Color digital images are used in many multimedia applications and in many vital applications. Some of these applications require excellent protection for these images because they are confidential or may contain confidential data. In this paper, a new method of data cryptography is introduced, tested, and implemented. It will be shown how this method will increase the security level and the throughput of the data cryptography process. The proposed method will use a secret image_key to generate necessary private keys for each byte of the data block. The proposed method will be compared with other standard methods of data cryptography to show how it will meet the requirements of excellent cryptography, by achieving the objectives: Confidentiality, Integrity, Non-repudiation, and Authentication.

Keywords:
Cryptography, image_key, block, PK, MSE, PSNR, throughput, speedup.

1. Introduction

Digital data, including color digital images [1-2], requires the protection process during its communication in order to prevent intruders or unauthorized persons from understanding this data or knowing its content for several reasons, including the fact that this data is confidential or private or carries confidential data [3]. The data protection process is carried out as shown in Figure 1 through an application of data cryptography, which means encrypting the data using a private key (PK) and implementing a specific method before sending the data and decrypting it using the same key and the same method after receiving the data. In order to prevent intruders from understanding the data, the encryption method must destroy the data so that it becomes incomprehensible or unreadable. As for the decryption process, it must return the sent data without changing or losing any part of the data [4-8].

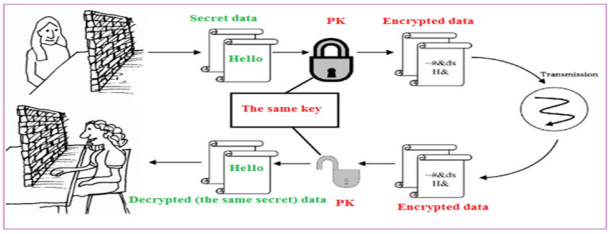


Fig. 1: Data Cryptography

The degree of data destruction is measured by quality factors MSE (mean square error) or/and PSNR (peak signal to noise ratio) between two data sets. The method of encryption and decryption is considered excellent if the value of the parameter PSNR is very low and the value of the parameter MSE is very high when encrypting and on the contrary when decrypting, the value of the parameter PSNR must be infinite and the value of the parameter MSE is zero [9-12].

The method of data cryptography that is chosen to protect the data should provide a high degree of security to prevent the process of penetration of confidential data, and this degree is reached through the use of folded and complex keys and the implementation of some operations that are difficult to guess. One of the important specifications that the data encryption process must achieve is its efficiency by minimizing the time required to implement the encryption process and the time required to implement the decryption process, and thus increasing the number of encrypted or decrypted bits per second (throughput). The selected method of data cryptography must be simple to implement and must require a minimum hardware environment.

The degree of protection provided by the encryption method depends largely on the secret key used and the mechanism for generating other sub-keys necessary to complete the encryption and decryption process, which leads us to the possibility of using the digital image to generate these keys for ease of processing digital images and the ease of retrieval of data from them. Its huge potential for what it provides because the digital color image provides a huge amount of data.

Manuscript received December 5, 2021
Manuscript revised December 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.12.89

The color image is represented by a 3D matrix, one 2D matrix for each color (red, green, and blue) as shown in Figure 2.

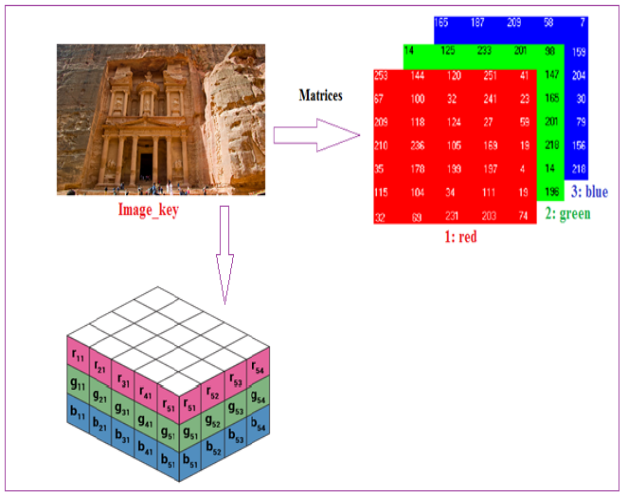


Fig. 2: Color image representation

The color digital image has many good features that can be employed to generate the private keys necessary for the data cryptography process, including:

- The digital image is widely available and can be obtained easily and at the lowest cost.
- The ability to deal with each color matrix separately or with parts of the image.
- The ability to resize the image size to generate data with one dimension and a specific size (see Figure 3).
- The possibility of forming the image and converting the data in it to a vertical or linear dimension (see Figure 4).

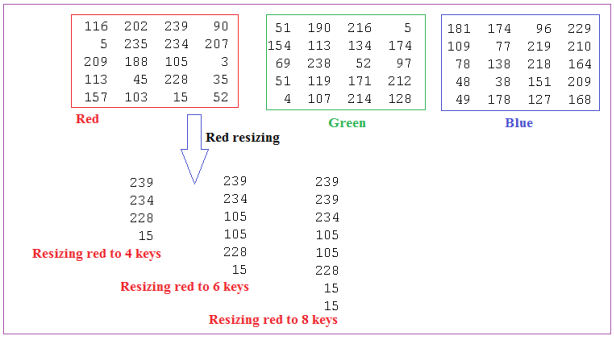


Fig. 3: Image resizing

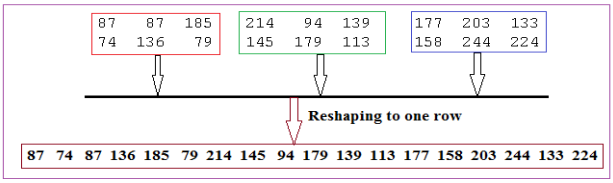


Fig. 4: Image reshaping

2. Related Work

Many methods are used to encrypt-decrypt confidential data [13-14], and many of these methods depend on some standards as a basis, the most important of which are [15], [13]:
Data encryption standard (DES), advance encryption standard (AES), triples DES (3DES), and blowfish method (BF).

Methods based on these standards have some conmen features (see table), the most of them are [16-19]:

- They use a PK with a fixed length, and sometimes it may be hacked.
- The data to be encrypted must be divided into a fixed number of blocks; each block has a fixed size.
- A set of routines are required to generate necessary sub-keys.
- A set of fixed rounds are needed to complete the encryption and decryption processes.
- A set of s-box is required which increases the amount of memory needed to implement the method.
- These methods are not efficient in color image encryption-decryption, and the encryption-decryption times will rapidly increase when the data size increases [19-23].

Table 1: Encryption-decryption methods main features

Algorithm parameter	DES	3DES	AES	Blowfish
PK length (bit)	56 (fixed)	112, 168 (fixed)	128, 192, 256 (fixed)	32-448 (fixed)
Block size (bit)	64 (fixed)	64 (fixed)	128 (fixed)	64 (fixed)

Ability to deal with images	Difficult	Difficult	Difficult	Difficult
Encryption Quality	Excellent: High MSE and low PSNR	Excellent: High MSE and low PSNR	Excellent: High MSE and low PSNR	Excellent: High MSE and low PSNR
Decryption Quality	Excellent: Zero MSE and infinite PSNR	Excellent: Zero MSE and infinite PSNR	Excellent: Zero MSE and infinite PSNR	Excellent: Zero MSE and infinite PSNR
Efficiency	Slow	Slow	Slow	Moderate
Attack	Brute force attack	Brute force attack, Known plaintext, Chosen plaintext	Side channel attack	Dictionary attack
Structure	Feistily	Feistel	Substitution-Permutation	Feistel
Block Cipher	Binary	Binary	Binary	Binary
Rounds	16 (fixed)	48 (fixed)	10,12,14(fixed)	16 (fixed)
Flexibility to modification	no	yes	yes	yes
Simplicity	no	no	no	no
Security level	Adequate	Adequate	Excellent	Excellent
Throughput	Low	low	Low	Moderate

3. Proposed Method

The one round variable block size (ORVBM) proposed method uses is based on using any color image with any size to generate the necessary keys for data cryptography and it can be considered as an improved method of data encryption by providing the following features:

- A color image_key is to be used to generate all needed keys.
- There is no restriction on the image size.
- The image key can be replaced at any time when needed without changing the method operations.
- ORVBM can be used to encrypt messages, text files, and digital images.
- The data to be encrypted can be divided into blocks with different block sizes (BLS).
- The number of keys will equal the block size in bytes.
- The contents and number of the key are varied and they are depending on the selected image_key and the block size (see tables 2 and 3).
- The Feistel function (FF) is simple and can be changed any time, this function rotates the byte 2 binary digits to the right when encryption and 6 binary digits when decryption.
-

The encryption phase of ORVBM as shown in Figure 5, can be implemented by applying the following steps:
Step 1: Select the image key.
Step 2: Select the data to be encrypted (image or text file) (a).
Step 3: Get the size of a.
Step 4: Get the block size.
Step 5: Reshape (a) into one row.
Step 6: Divide (a) into blocks.
Step 7: Get the keys by resizing the image key to the size of the reshaped (a).
Step 8: For each byte in the block do the following:

- Apply XOR with associate key.
- Apply Feistel function.

Step 9: Reshape the encrypted image back to the 3D matrix.

Table 2: Generated keys from image 4 (from the selected images in the implementation part)

BLS=4	BLS=6	BLS=7	BLS=10
Sub_private Keys	**Sub_private Keys**	**Sub_private Keys**	**Sub_private Keys**
122	141	156	107
122	147	123	144
160	157	148	147
212	165	165	150
K1 thru K4	231	155	154
	183	240	169
	K1 thru K6	175	151
		K1 thru K7	231
			221
			140
			K1 thru K10

Table 3: Generated keys from image 1 (from the selected images in the implementation part)

BLS=4	BLS=6	BLS=7	BLS=10
Sub_private Keys	**Sub_private Keys**	**Sub_private Keys**	**Sub_private Keys**
78	101	93	102
219	134	71	80
20	220	126	134
79	8	29	138

K1 thru K4	59	60	42
	86	52	38
	K1 thru K6	109	57
		K1 thru K7	59
			38
			132
			K1 thru K10

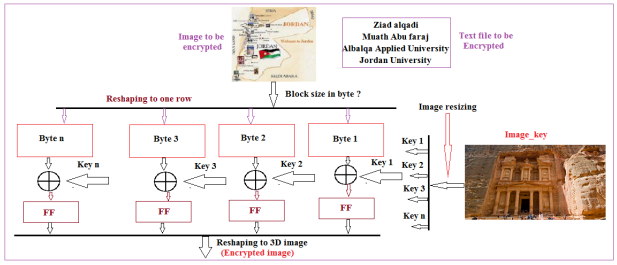


Fig. 5: Proposed ORVBM encryption phase

The decryption phase of ORVBM as shown in Figure 6 can be implemented by applying the following steps:
Step 1: Select the image key.
Step 2: Get the encrypted data (image or text file) (a).
Step 3: Get the size of a.
Step 4: Get the block size.
Step 5: Reshape (a) into one row.
Step 6: Divide (a) into blocks.
Step 7: Get the keys by resizing the image key to the size of the reshaped (a).
Step 8: For each byte in the block do the following:
- Apply Feistel function.
- Apply XOR with associate key.

Step 9: Reshape the image back to the 3D matrix.

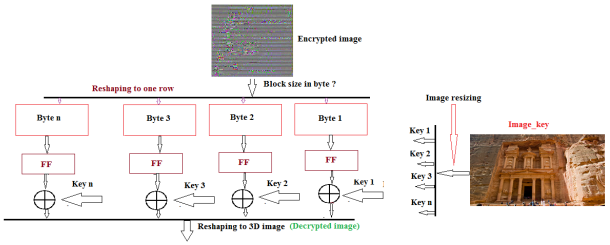


Fig. 6: PROPOSED ORVBM decryption phase

4. Implementation and Experimental Results

The proposed ORVBM and the other standard methods of data cryptography were programmed using Matlab, the programs were executed using a PC with i5 2.4 G Hz processor and 8 G byte RAM.

For an image block with 8 bytes and using the image_key shown in figures 9, figures 7 and 8 illustrate an example of encrypting a data block with block size =10:

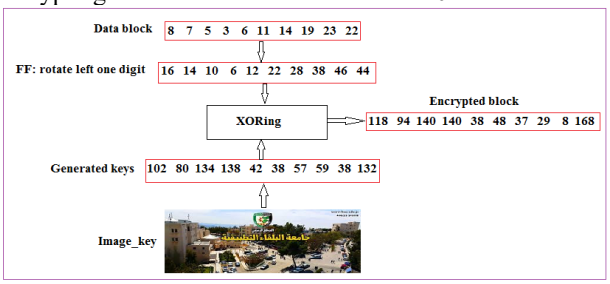


Fig. 7: Example of encrypting data block with 10 bytes

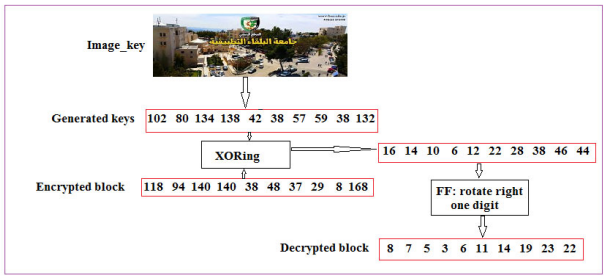


Fig. 8: Example of decrypting encrypted data block with 10 bytes

Fig. 9: Sample 1 output (block size=100 bytes)

Fig. 10: Sample 2 output (block size=100 bytes)

From Figures 9 and 10 we can see that both two different image_keys with different sizes gave good results

of MSE and PSNR values, but the second has better quality parameters values, so sometimes the values of MSE and PSNR depend on the selected image_key contents, but even the worst selected one will give good values for MSE and PSNR and we don't have to worry about the selected image to be used as a private key.

Table 4: Image encryption-decryption using image 4 as a key with block size =100 bytes

Image Number	Dimensions	Size (Byte)	Encryption Time (Seconds)	MSE	PSNR
1	151x333x3	150849	0.031000	1.1399e+004	17.4120
2	152x171x3	77976	0.017000	1.2013e+004	16.8877
3	360x480x3	518400	0.106000	1.1028e+004	17.7433
4	1071x1600x3	5140800	1.076000	1.0642e+004	18.0998
5	981x1470x3	4326210	0.887000	1.0633e+004	18.1083
6	165x247x3	122265	0.026000	1.0716e+004	18.0300
7	360x480x3	518400	0.108000	1.2170e+004	16.7583
8	183x275x3	150975	0.031000	1.1135e+004	17.6466
9	183x275x3	150975	0.031000	1.0495e+004	18.2386
10	201x251x3	151353	0.032000	1.1456e+004	17.3623
11	600x1050x3	1890000	0.391000	1.0569e+004	18.1684
12	1144x1783x3	6119256	1.262000	1.0793e+004	17.9586
Average		1609800	0.3332		
Throughput (byte/second)		4.8313e+006			
Throughput (Mbyte/second)		4.6075			

To show the effects of using a smaller image as an image_key the same experiment was repeated using a smaller image as an image_key; table 5 shows the obtained results :

Table 5: Image encryption-decryption using small image_key (image 2) with block size =100 bytes

Image Number	Dimensions	Size (Byte)	Encryption Time (Seconds)	MSE	PSNR
1	151x333x3	150849	0.037000	2.3889e+004	10.0133
2	152x171x3	77976	0.016000	2.9976e+004	7.7437
3	360x480x3	518400	0.108000	1.8635e+004	12.4974
4	1071x1600x3	5140800	1.059000	1.5115e+004	14.5906
5	981x1470x3	4326210	0.892000	1.4945e+004	14.7037
6	165x247x3	122265	0.026000	1.1781e+004	17.0833
7	360x480x3	518400	0.107000	2.1150e+004	11.2311
8	183x275x3	150975	0.041000	1.9196e+004	12.2008
9	183x275x3	150975	0.031000	1.4606e+004	14.9331
10	201x251x3	151353	0.032000	2.1849e+004	10.9060
11	600x1050x3	1890000	0.391000	1.9962e+004	11.8093
12	1144x1783x3	6119256	1.345000	8.0774e+003	20.8570
Average		1609800	0.3404		
Throughput (byte/second)		4618.3 K byte			
Throughput (Mbyte/second)		4.6183 M byte			

To show how the selected block size affects the quality and the performance of the proposed method, image 3 was taken and encrypted-decrypted using various block sizes and using image 4 as an image_key, table 6 shows the obtained experimental results.

Table 6: Using various block sizes to encrypt decrypt the same image

Block Size (Byte)	Number of PSK	Encryption Time (Seconds)	MSE	PSNR
8	8	0.015000	8.8013e+003	19.9987
10	10	0.017000	1.0455e+004	18.2765
18	18	0.025000	1.5942e+004	14.0580
32	32	0.040000	1.3409e+004	15.7885
64	64	0.071000	1.2620e+004	16.3948
128	128	0.136000	1.1064e+004	17.7109
200	200	0.222000	1.2119e+004	16.8000
256	256	0.264000	1.1268e+004	17.5283
400	400	0.436000	1.2650e+004	16.3714
512	512	0.517000	1.1351e+004	17.4543
1000	1000	0.998000	1.2779e+004	16.2696
2000	2000	2.031000	1.7309e+004	13.2357

The proposed ORVBM can be used to encrypt decrypt any image of any size; it is also can be used to encrypt-decrypt secret text files. A set of text files were selected and encrypted-decrypted using image 4 as an image_key with a block size equal to 100 bytes. The same text files were also encrypted-decrypted using the standard methods DES, 3DES, AES, and BF, Table 7 shows the obtained experimental results:

Table 7: Efficiency comparisons

Text File Size (Kbyte)	Encryption Time (Second)				
	DES	3DES	AES	BF	ORVBM
1	0.0043	0.0065	0.0033	0.0002	0.000012
10	0.0269	0.0296	0.0241	0.0137	0.001000
50	0.0499	0.0572	0.0463	0.0296	0.002000
100	0.0758	0.0858	0.0694	0.0409	0.003000
150	0.1237	0.1389	0.1137	0.0684	0.005000
200	0.1467	0.1676	0.1359	0.0807	0.006000
400	0.2893	0.3341	0.2698	0.1598	0.012000
500	0.3621	0.4169	0.3352	0.1997	0.015000
600	0.4340	0.4987	0.3996	0.2396	0.018000
800	0.5776	0.6653	0.5355	0.3199	0.024000
1000	0.7193	0.8289	0.6674	0.3996	0.030000
Average	0.2554	0.2936	0.2364	0.1411	0.0105
Throughput (Kbyte)	1357	1180	1466	2455	32996

5. Results Analysis

The proposed ORVBM provides a simple way to encrypt-decrypt any image with any size using any image_key; from the obtained experimental results we can raise the following important points:

1) ORVBM uses various block sizes, for each byte in the data block a key from an image_key will be extracted, this makes the private key complicated and hard to hack, thus it will increase the security level and will provide excellent protection for the secret digital images transmitted via a communication environment.
2) To increase the security level a secret Feistel function is added.
3) The encryption/decryption phases are accomplished using one round, this will decrease the encryption/decryption times, and thus the method throughput will increase.

4) The image_key must be kept secret and it can be replaced from time to time.
5) No matter what the block size is, ORVBM provides an excellent quality by maximizing MSE (and minimizing PSNR) in the encryption phase and providing zero MSE and infinite PSNR in the decryption phase.
6) The quality parameters MSE and PSNR do not affect by increasing or decreasing the image_key size (see Tables 4 and 5).
7) Decreasing the block size of the image to encrypted-decrypted will decrease the encryption-decryption times, thus it will increase the method efficiency (throughput), see Table 6, and Figure 11 shows the effects of increasing the image block size.

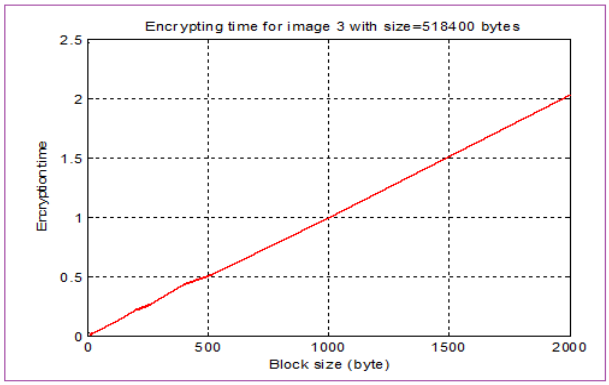


Fig. 11: Relationship between the block size and the method throughput

8) ORVBM can be used to encrypt-decrypt secret text files, doing this will improve the cryptography process efficiency, the comparison results with standard methods of data cryptography as shown in Table 7, from these results we can see that ORVBM will decrease the encryption-decryption time and this shown in Figure 12. From Tab le 7, we can see that ORVBM has a significant speedup compared with other methods, speedup of ORVBM will be calculated by dividing its throughput by the throughput of any other method, and this is shown in Table 8.

Table 8: Speedup calculation

Method	**DES**	**3DES**	**AES**	**BF**	ORVBM
DES	1.0000	1.1500	0.9256	0.5527	0.0411
3DES	0.8696	1.0000	0.8049	0.4807	0.0358
AES	1.0803	1.2424	1.0000	0.5971	0.0444
BF	1.8091	2.0805	1.6746	1.0000	0.0744
ORVBM	**24.3154**	**27.9627**	**22.5075**	**13.4403**	1.0000

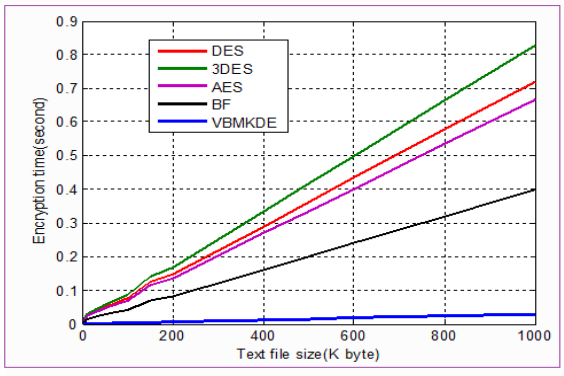


Fig. 12: Encryption times comparison

6. Conclusions

ORVBM of data cryptography was introduced and implemented. It was shown that this method can be used for images and text files cryptography. The image or the text file may have any size. ORVBM uses complicated private keys extracted from an image_key, the image_key is to be kept in secret and it can be replaced from time to time to ensure the security of the method. The data to be encrypted is to be divided into blocks with a secret size of bytes, for each byte a private key is to be assigned making the hacking process impossible. The proposed ORVBM provided

excellent values for quality parameters in both the encryption and decryption phases. ORVBM increased the efficiency of data cryptography and it was shown that it has a significant high speed up comparing with other standard methods used for data cryptography.

References

[1] Hamdan, M., Subaih, B., Alqadi, Z.: *Extracting Isolated Words from an Image of Text*. International Journal of Computer Science & Mobile Computing 5(11), 29-36 (2016)

[2] Hindi, A., Dr. Dwairi, M., Alqadi, Z.: Analysis of Procedures used to Build an Optimal Fingerprint Recognition System. International Journal of Computer Science and Mobile Computing 9(2), 21-37 (2020)

[3] Rushdi Abu Zneit, R., Al-Azzeh, J. Alqadi, Z., Ayyoub, B., Sharadqh, A.: *Using Color Image as a Stego-Media to Hide Short Secret Messages*. International Journal of Computer Science and Mobile Computing, vol. 8, issue 6, pp. 106-123, 2019.

[4] Patel, K.: *Performance analysis of AES, DES and Blowfish cryptographic algorithms on small and large data files*. International Journal of Information Technology 11, 81—819 (2019). https://doi.org/10.1007/s41870-018-0271-4

[5] Rani, R., Sharma, G.: *Review Paper on Data Hiding In 3D Barcode Image Using Steganography*. International Journal of Advanced Research in Computer Science 8(5), 2271-2276 (2017)

[6] Ghodke, M., Mali, N.: *FPGA Based Network Security Using Cryptography*. International Research Journal of Engineering and Technology 3(3):469-71 (2016)

[7] Bhuvaneshwari, M., Tenmozhi, S.: *A VLSI architecture for security based stenographic processor with AES algorithm*. International Journal of Electrical and Computer Engineering; 1–6.

[8] Sukhraliya, V., Chaudhary, S.: *Encryption and decryption using ASCII values with substitution array approach*. International Journal of Advanced Research in Computer and Communication Engineering. 2(8) 3094–3097 (2013)

[9] Shaikh, A. P, Kaul, V.: *Enhanced Security Algorithm using Hybrid Encryption and ECC*. IOSR Journal of Computer Engineering (IOSR-JCE) 16(3), 80-85 (2014)

[10] Garg, SK.:, *Modifed encryption and decryption using symmetric keys at two stages: Algorithm SKG 1.2*. International Journal of Advanced Research in Computer Science and Electronics Engineering. 4(6),778-80 (2014)

[11] Enriquez, M., Garcia, D. W., Arboleda, E.:, *Enhanced Hybrid Algorithm of Secure and Fast Chaos-based, AES, RSA and ElGamal Cryptosystems*. Indian Journal of Science and Technology 10(27), 1-14 (2017) DOI: 10.17485/ijst/2017/v10i27/105001.

[12] Vijayalakshmi, C., Lavanya, L., Navya, C.: *A Hybrid Encryption Algorithm Based On AES and RSA*. International Journal of Innovative Research in Computer and Communication Engineering 4(1), 909-917 (2016)

[13] Li, P., Chen, Z., Yang, L. T., Zhao, L. , and Zhang, Q.: *A Privacy-preserving High-order Neuro-fuzzy C-means Algorithm with Cloud Computing*. Neurocomputing 256, pp. 82–89 (2017)

[14] Yin, S., and Liu, J.: *A K-means Approach for Map-reduce Model and Social Network Privacy Protection*. Journal of Information Hiding and Multimedia Signal Processing 7(6), 1215-1221 (2016)

[15] Zhang, Q., Yang, L. T., Chen, L. T. and Li, P.: *High-order Possibilistic c-means Algorithms Based on Tensor Decompositions for Big Data in IoT*. Information Fusion 39, 72–80 (2018)

[16] Teng, L., Li, H., Liu, J., and Yin, S.: *An Efficient and Secure Cipher-text Retrieval Scheme Based on Mixed Homomorphic Encryption and Multi-attribute Sorting Method.* International Journal of Network Security 20(5), 872-878 (2018)

[17] Elgendy, I., Zhang, W., Liu, C., and Hsu, C.: An Efficient and Secured Framework for Mobile Cloud Computing. IEEE Transactions on Cloud Computing 9(1), 79-87 (2018)

[18] B. Karthikeyan, T. Sasikala, and S. B. Priya, "Key exchange techniques based on secured energy efficiency in mobile cloud computing," *Applied Mathematics & Information Sciences*, vol. 13, no. 6, pp. 1039–1045, 2019.

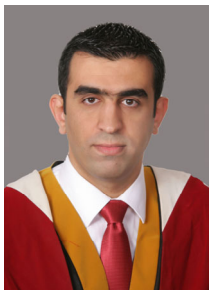
[19] Xu, J. Wei, L., Wu, W., Wang, A., Zhang, Y., and Zhou ,F.: *Privacy-preserving Data Integrity Verification by using Lightweight Streaming Authenticated Data Structures for Healthcare Cyber-physical System*. Future Generation Computer Systems 108, pp. 1287-1296 (2020)

[20] Haseeb, K., Almgren, A., Ud Din, I., Islam, N., and Altameem, A.: *SASC: Secure and Authentication-based Sensor Cloud architecture for intelligent Internet of Things*. Sensors 20(9), pp. 2468-2486 (2020)

[21] Zhu, H., Yuan, Y., Chen, Y., et al.: *A Secure and Efficient Data Integrity Verification Scheme for Cloud-IoT Based on Short Signature*. IEEE Access 7, pp.9003-90044 (2019)

[22] Zhang, Z., and Luo, J.: *A Data Value Classification and Encryption Mechanism Based on Metadata Attributes.* Journal of Northwest University 46(2) pp. 188-194 (2016)

[23] Liu, T., Liu, Y. Mao, Y. et al.: *A Dynamic Secret-based Encryption Scheme for Smart Grid Wireless Communication*. IEEE Transactions on Smart Grid 5(3), 1175-1182 (2014).



Mua'ad Abu-Faraj received the B.Eng. degree in Computer Engineering from Mu'tah University, Mu'tah, Jordan, in 2004, the M.Sc. degree in Computer and Network Engineering from Sheffield Hallam University, Sheffield, UK, in 2005, and the M.Sc. and Ph.D. degrees in Computer Science and Engineering from the University of Connecticut, Storrs, Connecticut, USA, in 2012. He is, at present, an Associate Professor at The University of Jordan, Aqaba, Jordan. He is currently serving as a reviewer for the IEEE Micro, IEEE Transactions on Computers, Journal of Supercomputing, and International Journal of Computers and Their Applications (IJCA). His research interests include computer architecture, reconfigurable hardware, image

processing, cryptography, and wireless networking. Dr. Abu-Faraj is a member of the IEEE, ISCA (International Society of Computers and their Applications), and JEA (Jordan Engineers Association).

Ziad A. Alqadi received the B.E., M. E., and Dr. Eng. degrees from Kiev Polytechnic Institute. in 1980, 1983, and 1986, respectively. After working as, a researcher from 1986, an assistant professor from 1991 in the department of Electrical Engineering, Amman Applied College, and an Associate Professor from 1996 in the Faculty of Engineering Technology, he has been a professor at Albalqa Applied. since 2010. His research interest includes signal processing, image processing, data security, and parallel processing.