

A computational enhancement of Base64 algorithm using Residue Number System algorithms

Kolawole Bariu LOGUNLEKO^{1*}, Abolore Muhamin LOGUNLEKO², Ayisat Wuraola ASAJU-GBOLAGADE³, Akinbowale Nathaniel BABATUNDE⁴ & Kazeem Alagbe GBOLAGADE⁵

***Corresponding Author:**

Kolawole Bariu LOGUNLEKO
kolawolelogunleko@gmail.com

^{1*} Department of Computer Science, D.S. Adegbenro ICT Polytechnic,
Eruku-Itori, Ewekoro, Ogun state, Nigeria

² Department of Computer Science, Gateway ICT Polytechnic, Saapade, Ogun State, Nigeria

³ Department of Computer Science, Faculty of Communication and Information Sciences,
University of Ilorin, Ilorin, Nigeria

^{4,5} Department of Computer Science, Kwara State University, Malete, Kwara State, Nigeria
kolawolelogunleko@gmail.com, aboloremlogunleko@gmail.com, ayisatwuraola@gmail.com,
akinbowale.babatunde@kwasu.edu.ng, kazeem.gbologade@kwasu.edu.ng

Abstract: Today's daily existence depends heavily on the advancement of data, information and communication technologies. Consequently, there is a huge increase in the need for data and information. In communicating data over a public network, data security is a necessity that must be carefully taken into consideration. Thus, Base64 algorithm has been used in numerous security applications for ensuring data confidentiality, integrity and authentication. However, research shows that there are security vulnerabilities in the most widely used Base64 algorithm due to the absence of a key mechanism. To address this concern, the research employs residue number system algorithms for the enhancement of the Base64 algorithm because of its cryptographic features and thereby strengthens the transformation of the existing Base64 algorithm to produce a novel symmetric-based cryptographic algorithm. The developed algorithm generates a symmetric key by shuffling the original key with the textual data, making the transformation of each character of the data better each time it is shuffled. Therefore, the research bridges the security gap in the Base64 cryptographic algorithm by factoring the key mechanism of the RNS based algorithm into the newly developed algorithm. The pattern and confusion produced during the methodology procedure safeguards the data more effectively as shown in the cipher text generated.

Keywords: Base64 Algorithm, Key, Residue Number System Algorithm, Xor Operation.

1. Introduction

The increased use of the internet, the availability of the digital data and the dissemination of data have made information security a crucial area for academics and professionals in information technology (Eseyin & Gbolagade, 2019). The development of the information technology has brought about a number of important changes in people's lives, including a paradigm shift that gave rise to new forms of wealth and the reality that data is now replacing oil (Agusta, 2021). As time goes by, people's need for data and information are increasing tremendously therefore, the security of data, information and communication technology are currently an essential part of everyday human life (Baso, 2023).

According to Logunleko et al. (2020), the privacy of data must be carefully taken into account before being sent or shared over a public network. Thus, users need to take the necessary precautions by enforcing some protective measures on the information to be transmitted. For data security, encryption is the most significant automated technique (Victor et al., 2023). Only a communication route shielded by encryption is protected from prying eyes attempting to steal

private information. Furthermore, the era of information and communication technology demands that the communication sector thrive since security is a crucial issue that needs to be resolved. In order to support the data security measures in guaranteeing data privacy, secrecy and authentication, among other things, the Base64 cryptographic algorithm is employed. Several security applications have employed the Base64 algorithm to guarantee data integrity and authenticate the source of data but research found that the algorithm is not adequately secured due to the absence of a key mechanism. Moreover, the main goals in developing an algorithmic encryption and decryption process are to improve the security and efficiency of the Base64 algorithm. Thus, this research integrates residue number system algorithms into the Base64 algorithm to produce a symmetric-based cryptographic algorithm called residue number system-Base64 algorithm (RNS-B64) in order to ensure a better and more secured cryptosystem.

The paper is structured as follows: related papers are discussed in Section 2. Section 3 focuses on the approach while in Section 4 and 5 are the results and the conclusion, respectively.

2. Related work

A number of studies (Sumartono, Siahaan & Arpan, 2016; Sarath et al., 2021; Efendi, Sihombing & Parulian, 2022) developed Base64 Algorithms (as stated in Algorithm1 and 2) and operation procedures in details, the first stage dividing the three binary data bytes (24 bits) into four numbers of six bits each. Base64 utilizes 6 bits, i.e. $2^6 = 64$ characters, to guarantee that the encoded data is legible and does not employ any of the special characters present in ASCII, despite the ASCII standard requiring the usage of seven bits. These 64 ASCII characters are used in the base64 algorithm as shown in Figure1.

The algorithm 1 transforms ASCII code into an 8-bit binary string, then adds the 8-bit binary together and divides it into 6 bits, converts them to decimals and selects a character element from Base64 lookup table. Conversely, algorithm 2 involves locating each character of the cipher text in the Base64 lookup table, creating a 6-bit binary, combining the binary strings, dividing them into eight-bit strings, converting them to decimals, thereafter equivalent Base64 character is obtained from Base64 lookup table.

Algorithm 1: Encryption Algorithm

```

bi = ascii(mi)
bi = bin(bi)
ci = split(bi, 8)
c = concat(c0, c1, ..., cn)
ci = split(c, 6)
ci = base64(ci)

```

Algorithm 2: Decryption Algorithm

```

ci = base64'(ci)
bi = join(ci)
bi = bin'(bi)
ci = split(bi, 6)
c = concat(c0, c1, ..., cn)
ci = split(c, 8)
mi = ascii'(ci)

```

Base64 Encoding/Decoding Table															
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Q	R	S	T	U	V	W	X	Y	Z	a	b	c	d	e	f
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
w	x	y	z	0	1	2	3	4	5	6	7	8	9	+	/
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63

Figure1. Base64 Encoding/Decoding Lookup Table

Source: (Sarath et al., 2021)

Baso (2023) applied Base64 algorithm to image files so as to ensure the security of data from unauthorized users. The researcher revealed that Base64 was often used for simple encryption purposes such as hiding non-sensitive information and authentication processes. However, the study further pointed out that Base64 algorithm had a security inadequacy mechanism because of its no key technique. Therefore, the algorithms' security needs to be improved so as to make the algorithm stronger for a symmetric based cryptography.

Asaju, Popoola & Gbolagade (2022) did a research on the enhancement of the image security with RNS as DES is susceptible to brute force attack. After applying watermarking, the resulting model needed a secret key and RNS moduli set to retrieve the original image. By watermarking one image inside another, the watermarking technique was able to give an initial level of image protection, even confusing the original image for an impartial user. The computing time and the memory space required to execute each of the algorithms under consideration were utilized to test and evaluate the model. The obtained results show that the residual number did not only add another layer of security, but also helped the image to be processed faster during the encryption and decryption due to the fact that only a residual part of the image was worked upon rather than the whole image pixels.

Naser (2021) explained cryptography as the modern security protocol to protect the information from the outsiders and to communicate securely without involving third parties. In addition to introducing a generic way of solvability series for a solvable problem that differs from others, the researcher suggested a few modified solvability propositions. Thus, a new basic numerical model of cryptography was developed based on solvability series.

Stanislaw et al. (2022) proposed a novel symmetric cryptography technique for the Residue Number System and its Modified Perfect Form. In the first technique, the ciphertext is considered as a collection of residues to the corresponding sets of modules (keys), and the Chinese Remainder Theorem was used to determine the decimal number recovery from its residues. The amount of arithmetic operations required for the decryption procedure was reduced as a result of the authors' use of a Modified Perfect Form of Residue Number System to streamline the computations. In a similar vein, where quick decryption was necessary, the study used a different symmetric encryption technique based on the Chinese Remainder Theorem. The Prime Number Theorem and the Euler Function were used to assess the security of the suggested techniques. The study examined the bitness and number of modules needed to provide the same level of security for the created symmetric security systems as the AES algorithm's longest length key. The study also, discovered that the bitness of the modules reduces with an increase in number.

Aremu & Gbolagade (2017) presented the design methods of information encryption and decryption using Residue Number System (RNS). The study selected three moduli set

$\{2^n - 1, 2^n, 2^n + 1\}$ and design an effective forward conversion for the selected moduli set with $4n + 2$ and $8n$ as delay and Area respectively, for the information encryption and reverse converter for the same moduli set with $4n + 3$ and $8n + 3$ as area and delay respectively, the proposed scheme outperforming the state of the art in terms of both security and computational efficiency.

Babatunde & Oloyede (2021) applied RNS and MPEG IV algorithm in video security. The work presents a cryptographic scheme for enhancing video transmission using a proposed modification of the traditional moduli set $\{2^n - 1, 2^n, 2^n + 1\}$.

Thiagarajan et al. (2018) proposed an encryption and decryption algorithm using the algebraic matrix approach as an efficient data algorithm to protect the message with the help of a key passed between the sender and the receiver. Utilizing the advantages of message splitting to an amount of words, the data encryption lowered the computational and storage overheads for the data owner. The ease of use and accessibility of the created model demonstrated that tools may be created without having to buy pricey software off the shelf.

The study revealed that the Base64 algorithm had been used in numerous security applications for ensuring data confidentiality and integrity. However, the Base64 transformation cryptography algorithm needed to be improved upon due to its non-availability of key. A cryptographic scheme's level of security is largely determined by the type and length of keys used, the degree of encryption used to create chaos, the algorithms' throughput rate, and their capacity to encrypt shorter messages (Stinson, Paterson & Maura 2019). Based on this, the Base64 algorithm is therefore enhanced using Residue Number System due to its cryptographic features thereby creating more confusion and transformation to the crypto algorithms to produce the RNS-Base64 cryptosystem.

3. Methodology

This study develops an enhanced version of the Base64 algorithm by factoring the key into the algorithm using the residue number system (RNS) approach. The purpose of the developed algorithm is to boost the efficiency and performance of the Base64 algorithm current capabilities and also to increase the algorithm security level. The addition of the use of the restricted moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ helps strengthen further the security of the Base64 algorithm to form RNS-B64 Algorithm.

3.1. Residue Number System

A Residue Number System (RNS) uses a collection of smaller integers to represent a large number so as to make computations more efficient. RNS functions in accordance with the Sunzi Suanjing, a mathematical concept that was developed in the fourth century AD and is based on the Chinese remainder theorem of modular arithmetic. By definition, residue arithmetic operations such as addition, subtraction and multiplication are carry-free as each result digit is a function of just one digit from each operand and is hence independent of all other digits (Akanni, Eseyin & Gbolagade, 2022). RNS is extremely good for a wide range of applications, including digital signal processing, communications engineering, cybersecurity (cryptography), image processing, speech processing, and transformation, where addition and multiplication are crucial arithmetic operations (Singh, 2008).

3.2. RNS-Base64 algorithm

The developed algorithm accepts inputs such as key, message and ciphertext which are transformed to ASCII value. Thereafter, RNS was integrated into the ASCII value which was segmented into 8bits and transformed to Base64 Equivalent using Base64 Algorithm to form a symmetric encryption and decryption algorithms as shown in algorithm 3 and 4 respectively.

Algorithm 3: Encryption Algorithm

$$k_i = \text{ascii}(k_i)$$

$$f_i = (k_i * \text{len}(k)) / \text{count}(k_i)$$

$$d = \text{xor}(f_0, f_1, \dots, f_n)$$

$$m_i = \text{rns}(d)$$

$$m = \text{xor}(m_0, m_1, \dots, m_n)$$

$$b = \text{bin}(m)$$

$$m_i = \text{ascii}(m_i)$$

$$r_{ni}, r_{ni+1}, r_{ni+2} = \text{rns}(m_i)$$

$$r_i = \text{bin}(r_i)$$

$$r_i = \text{xor}(r_i, b)$$

$$r = \text{concat}(r_0, r_1, \dots, r_n)$$

$$c_i = r_{[6i:6i+6]}$$

$$ci = \text{base64}(c_i)$$

Algorithm 4: Decryption Algorithm

$$k_i = \text{ascii}(k_i)$$

$$f_i = (k_i * \text{len}(k)) / \text{count}(k_i)$$

$$d = \text{xor}(f_0, f_1, \dots, f_n)$$

$$m_i = \text{rns}(d)$$

$$m = \text{xor}(m_0, m_1, \dots, m_n)$$

$$b = \text{bin}(m)$$

$$c_i = \text{base64}^{-1}(c_i)$$

$$c = \text{concat}(c_0, c_1, \dots, c_n)$$

$$r_i = c_{[8i:8i+8]}$$

$$r_i = \text{xor}(r_i, b)$$

$$r_i = \text{dec}(r_i)$$

$$m_i = \text{crt}(r_{ni}, r_{ni+1}, r_{ni+2})$$

$$m_i = \text{ascii}(m_i)$$

Encryption Algorithm: The algorithm 3 is explained as follows:

Key

Use ASCII to convert to decimal

Provide the weight function to the decimal number ($f_i = (k_i * \text{character count}) / \text{weight}$)

XOR each decimal key

Utilizing the given moduli set to compute residue

XORing residues together

Convert the output to 8-bit binary

The message

Use the ASCII table to find characters

Use the provided modulus to calculate each decimal's remainders

Convert all decimal remainders to binary data in 8 bits

XOR each message's 8-bit binary with the key's 8-bit binary

Combine all binary 8-bit data

Dividing into 6-bit groups

Converting each group using a Base64 lookup table

Decryption algorithm: The algorithm 4 is explained as follows:

Repeat the key mechanism procedure in algorithm 3

Ciphertext

Use the Base64 lookup table to get the corresponding 6-bit binary for each character.

Combine all binary 6-bit

Divide merge into 8-bit groups
 XOR each 8-bit ciphertext binary with the 8-bit key binary
 Decimalize the result
 Use the provided modulus and n consecutive decimal integers to compute the original number with CRT, where n is the number of modulus supplied.
 Search the ASCII table for each calculated decimal.

4. Results and discussions

4.1. Base64 algorithm evaluation

Supplying the plaintext "Names" having five characters. The illustration in Table 1 shows the steps of the existing Base64 algorithmic process design.

Table1. Base64 Algorithm Illustration

Index	1	2	3	4	5
char	N	a	m	e	s
Decimal	78	97	109	101	115

The computational evaluation process begins:

Index 1	:	N	ASCII	:	78	Binary	:	0 1001110
Index 2	:	a	ASCII	:	97	Binary	:	01100001
Index 3	:	m	ASCII	:	109	Binary	:	0 1101101
Index 4	:	e	ASCII	:	101	Binary	:	0 1100101
Index 5	:	s	ASCII	:	115	Binary	:	0 1110011

Putting the binaries together gives

0 1001110011000010 11011010 11001010 111001100000000.

The combined length of the concatenated binaries is 40 bits, so in order to make it 48 bits, a multiple of 24 bits and 8-bit padding methods were required. Because of this, the combined binaries became

0 1001110011000010 1101101 || 0 11001010 111001100000000.

Eight 6-bit binaries would be created from this concatenated 48-bit, one for each grouping.

0 10011 || 100110 ||00010 1|| 101101|| 0 11001 || 010 111|| 001100|| 000000.

Next, each 6-bit binary will be transformed to a decimal value as seen in Table 2.

Table 2. Binary 6-Bit

Index	1	2	3	4	5	6	7	8
Binary 6-bit	0 10011	100110	00010 1	101101	0 11001	010 111	001100	000000
Decimal Value	19	38	5	45	25	23	12	0

After reviewing the decimal equivalent values for each index in the base64 lookup table, it is found that the plaintext "Names" converted into the ciphertext TmFtZXMA. As A denotes zero (0), it was substituted with the "=" sign, ultimately resulting in **TmFtZXM=** as the ciphertext.

From the computational analysis above, the amount of the plain text character is five (5), this result to a total of forty eight (48) bits emanating from **6 × 8 bits** as a result of an 8-bit padding mechanism. Thus, the 48-bits later grouped into 6-bits characters to produce cipher text of eight (8) characters.



4.2. RNS-Base64 algorithm evaluation

The RNS-Base64 Algorithm is evaluated computationally in this section. The section is carried out theoretically using the moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ for both the key introduced and the textual data. The section contains two parts: key and the message respectively.

Step 1: Key Generation

This involves the key calculation which entails the computational technique of the key feature using RNS concept with the restricted moduli set of $\{2^n - 1, 2^n, 2^n + 1\}$. This is narrated in Table 3.

Table 3. Key Generation

Index	Char	Decimal	Weight	Key Function	 Xor	Secret RNS	 Xor	Binary
1	K	75	1	300	1916	{25,28,2}	$25 \oplus 28 \oplus 2 = 7$	00000111
2	o	111	1	444				
3	l	108	1	864				
4	a	97	1	1164				

From Table 3, the characters used for the key is "Kola". Each character is converted to a decimal equivalent. Then, the key function is applied to each character and all the key function for each character are xored together. Furthermore, the secret RNS modulo set generated from the restricted moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ is applied which then xored to produce 7, finally, the binary equivalent of 8 bits is obtained for the process as shown in Table 3.

Step 2: Encryption Generation

Encryption Generation

The techniques of the plaintext "Names" begins. This contains the computational evaluation technique of encryption generation using RNS-Base64 Algorithm with restricted moduli set to generate eight-bit binaries as shown in Table 4.

Table 4. Transformation of Decimal Value to Binary Equivalent

Character	Decimal Value	Secret RNS	Binary		
N	78	{16, 14, 12}	00010000	00001110	00001100
a	97	{4, 1, 31}	00000100	00000001	00011111
m	109	{16, 13, 10}	00010000	00001101	00001010
e	101	{8, 5, 2}	00001000	00000101	00000010
s	115	{22, 19, 16}	00010110	00010011	00010000

Step 3: Xoring the plaintext binary with the key

Binary:

$$00010000 \oplus 00000111 = 00010111$$

$$00001110 \oplus 00000111 = 00001001$$

$$00001100 \oplus 00000111 = 00001011$$

Binary:

$$00000100 \oplus 00000111 = 00000011$$

$$00000001 \oplus 00000111 = 00000110$$

$$00011111 \oplus 00000111 = 00011000$$

Binary:

$$00010000 \oplus 00000111 = 00010111$$

$$00001101 \oplus 00000111 = 00001010$$

$$00001010 \oplus 00000111 = 00001101$$

Binary:

$$00001000 \oplus 00000111 = 00001111$$

$$00000101 \oplus 00000111 = 00000010$$

$$00000010 \oplus 00000111 = 00000101$$

Binary:

$$00010110 \oplus 00000111 = 00010001$$

$$00010011 \oplus 00000111 = 00010100$$

$$00010000 \oplus 00000111 = 00010111$$

Step 4: Concatenating the result binaries in step 3, the following binary sequences were generated.

0001011100001001000010110000001100000110000110000001011100001010
00001101000011110000001000000101000100010001010000010111

Step 5: Here, splitting Step 4 into six-bit binary, the following binary sequences were obtained.

000101 110000 100100 001011 000000 110000 011000 011000 000101 110000
101000 001101 000011 110000 001000 000101 000100 010001 010000 010111

Step 6: In this step, the Six-bit binary sequences obtained from Step 5, are then converted each to decimal value and the Base64 equivalent is generated respectively. This is demonstrated and summarized in Table 5.

Table 5. Transformation of Binary Value to Base64 Equivalent

S/NO	1	2	3	4	5	6	7	8	9	10
BINARY	000101	110000	100100	001011	000000	110000	011000	011000	000101	110000
DECIMAL VALUE	5	48	36	11	0	48	24	24	5	48
BASE 64 EQUIVALENT	F	w	k	L	A	w	Y	Y	F	w
S/NO	11	12	13	14	15	16	17	18	19	20
BINARY	101000	001101	000011	110000	001000	000101	000100	010001	010000	010111
DECIMAL VALUE	40	13	3	48	8	5	4	17	16	23
BASE 64 EQUIVALENT	o	N	D	w	l	F	E	R	Q	X

Cypher text: FwkLAWYYFwoNDwIFERQX is generated from the RNS-B64 algorithms.

The process of the decryption generation is as follows:

Step 1: Perform the key-generating procedure in Table 3 again.

Step 2: Using the matching 6-bit binary to replace "FwkLAWYYFwoNDwIFERQX"

000101 110000 100100 001011 000000 110000 011000 011000 000101 110000
101000 001101 000011 110000 001000 000101 000100 010001 010000 010111

Step 3: Combining the result binary in step 2

00010111 00001001 00001011 0000001100000110 000110000001011100001010
00001101000011110000001000000101000100010001010000010111

Step 4: Division of the step 3 binary into eight bits.

00010111 00001001 00001011 00000011 00000110 00011000 00010111 00001010
00001101 00001111 00000010 00000101 00010001 00010100 00010111

Step 5: Xoring the cipher-text binary and the key binary respectively.

Binary:

$$00010111 \oplus 00000111 = 00010000$$

$$00001001 \oplus 00000111 = 00001110$$

$$00001011 \oplus 00000111 = 00001100$$

Binary:

$$00000011 \oplus 00000111 = 00000100$$

$$00000110 \oplus 00000111 = 00000001$$

$$00011000 \oplus 00000111 = 00011111$$

Binary:

$$00010111 \oplus 00000111 = 00010000$$

$$00001010 \oplus 00000111 = 00001101$$

$$00001101 \oplus 00000111 = 00001010$$

Binary:

$$00001111 \oplus 00000111 = 00001000$$

$$00000010 \oplus 00000111 = 00000101$$

$$00000101 \oplus 00000111 = 00000010$$

Binary:

$$00010001 \oplus 00000111 = 00010110$$

$$00010100 \oplus 00000111 = 00010011$$

$$00010111 \oplus 00000111 = 00010000$$

Step 6: The eight-bit binary sequences obtained from step 5, are then converted back using the Chinese Remainder Theorem (CRT) to produce their decimal value, thereby generating the equivalent initial character. The computational method in Table 6 is achieved as a result of the backward conversion of RNS.

Table 6. Transformation of Binary Value to Character Equivalent

Binary	CRT(Secret RNS)	Decimal Value	Character
00010000 00001110 00001100	{16,14,12}	78	N
00000100 00000001 00011111	{4,1,31}	97	a
00010000 00001101 00001010	{16,13,10}	109	m
00001000 00000101 00000010	{8,5,2}	101	e
00010110 00010011 00010000	{22,19,16}	115	s

Consequently, the plaintext 'Names' is obtained. From the evaluation, the amount of the plain text character is five (5), this result to a total of one hundred and twenty (120) bits emanating from 8×15 bits (i.e. the 120 bits are grouped into fifteen (15) sections of 8-bits characters). Therefore, the 120-bits later grouped into 2-bits characters to produce the non-intelligible text referred to cipher text of Twenty (20) characters. Finally, the equivalent character of each of the above decimal from the ASCII table is **Names** which was initial plaintext, as the final output of the decrypted process.

5. Conclusion

The research revealed the essential application usage of the Base64 algorithm in information security and also pointed out the security gap in the existing Base64 cryptographic algorithm as it is not adequately secured. Therefore, the study employed RNS because of its cryptographic nature and efficiency peculiarity to enhance Base64 algorithm in order to produce a better symmetric key-based cryptographic algorithm. Furthermore, the addition of the use of the moduli set also strengthens further the security of the developed algorithm. The algorithm pattern established during the process of methodology procedure safeguards the data more effectively than the existing Base64 algorithm as it is observed in the cipher text generated. The developed RNS-Base64 algorithm produced a greater confusion and adequately secured in the hands of an adversary. The conducted computational analysis comes to the conclusion that, the research closes the security gap in the Base64 cryptographic algorithm as compared to the developed Residue Number System-Base64 cryptographic algorithms. Therefore, the developed RNS-B64 algorithm is theoretically and computationally secure, and it is suitable for exploration in information security systems for both encryption and decryption.

REFERENCES

- Agusta, H. (2021) Keamanan dan Akses Data Pribadi Penerima Pinjaman Dalam Peer to Peer Lending di Indonesia. *Krtha Bhayangkara*. 15(1), 11–38. doi: 10.31599/krtha.v15i1.289.
- Akanni, G. A., Eseyin, J. B. & Gbolagade, K. A (2022) A Residue Number System and Secret Key Crypto System Review in Cyber Security. 7(10), 1896-1901. doi: 10.5281/zenodo.7334218.
- Aremu, I. A. & Gbolagade, K. A. (2017) Information encoding and decoding using Residue Number System for $\{2^{2n} - 1, 2^{2n}, 2^{2n+1}\}$ moduli sets. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*. 6(8), 1260–1267.
- Asaju, B., Popoola, D. & Gbolagade, K. (2022) Enhancing Image Security Using Data Encryption Standard, Discrete Wavelet Transform Watermarking, Residue Number System and Gaussian Filtering. *African Scholar Journal of African Innovation & Advanced Studies*. 25(2), 19–40.
- Babatunde, A. N. & Oloyede, A. A. (2021) Application of Residue Number Systems in enhancing the transmission of secured videos. *Revista Română de Informatică Și Automatică [Romanian Journal of Information Technology and Automatic Control]*. 31(4), 97-108. doi: 10.33436/v31i4y202108.
- Baso, F. (2023) Analysis and Utilization of the Base64 Algorithm for Image Encryption and Decryption Security in Web-Based Images. *Journal of Security, Computer, Information, Embedded, Network, and Intelligence System*. 1(1), 52–57. doi: /10.61220/scientist.v1i2.20233.
- Efendi, M., Sihombing, V. & Parulian, S. (2022) Implementation and Use of Base64 Algorithm in Video File Security. *Sinkron*. 7(1), 243-247. doi:10.33395/sinkron.v7i1.11256.
- Eseyin, J. & Gbolagade, K. A. (2019) A Residue Number System Based Data Hiding Using Steganography and Cryptography. *Kampala International University (KIU), KIU Journal of Social Sciences*. 5(2), 345–351.
- Logunleko, K. B., Adeniji, O. D., Logunleko, A. M. & Odufowora, M. O. (2020) Data Encryption Scheme Using An Enhanced Base64 Algorithm. University of Ibadan. *Journal of Science and Logistics in ICT Research (UIJSLICTR)*. 3(1), 16–24.
- Naser, S. M. (2021) Cryptography: From the Ancient History to now, It's applications and a new complete numerical model. *International Journal of Mathematics and Statistics Studies*. 9(3), 11–30.
- Sarath, C., Sugadev. S, Chandragandhi, S., Akshai K. & Kamalesh M. (2021) Secure Message Transmission Using Base 64 Algorithm. *International Advanced Research Journal in Science, Engineering and Technology*. 08(04), 52–55. doi: 10.17148/IARJSET.2021.8411.

Singh, N. (2008) An overview of residue number system. *National Seminar on Devices, Circuits & Communication Organized by Department of ECE, B.I.T, Mesra, Ranchi-* 835215.

Stanislaw, Z., Mykhailo, K., Igor, I. & Daniel, J. (2022) Methods of crypto-stable symmetric encryption in the residual number system. *Elsevier*. 207, 128–137. doi:10.1016/j.procs.2022.09.045.

Stinson, D. R., Paterson, M. B. & Paterson, M. P. (2019) *Cryptography: Theory and Practice*. CRC Press Taylor & Francis Group LLC, 6000 Broken Sound Parkway NW, Suite 300, Boca Raton London New York, FL 33487-2742. <https://taylorandfrancis.com/>.

Sumartono, I., Siahaan, A. P. U. & Arpan. A. (2016) Base64 Character Encoding and Decoding Modeling. *International Journal of Recent Trends in Engineering & Research (IJRTER)*. 02(12), 63–68. <https://www.researchgate.net/publication/311715821>.

Thiagarajan, K., Balasubramanian, P., Nagaraj, J. & Padmashree, J. (2018) Encryption and decryption algorithm using algebraic matrix approach. *Journal of Physics: Conference Series, 1000*, 012148. doi: 10.1088/1742-6596/1000/1/012148.

Victor, M., Praveenraj, D. D. W., R, S., Alkhayat, A. & Shakhzoda, A. (2023) Cryptography: Advances in Secure Communication and Data Protection. *E3S Web of Conferences*. 399, 07010. doi: 10.1051/e3sconf/202339907010.



Kolawole Bariu LOGUNLEKO is a Lecturer at the Department of Computer Science, D.S. Adegbenro ICT Polytechnic Eruku-Itori, Ewekoro, Ogun State, Nigeria. He bagged his B.Sc (Hons) Degree in Mathematical Sciences (Computer Science Option) with Second Class Upper Division at the Federal University of Agriculture, Abeokuta, Ogun State, Nigeria and his Master Degree in Computer Science at the prestigious Premier University, University of Ibadan, Oyo State, Nigeria. At present, he is a Ph.D. student at the Department of Computer Science, Kwara State University, Malete, Kwara State, Nigeria. Besides, he is an active member of the Computer Professionals (Registration Council of Nigeria), Nigeria Computer Society, Academia in Information Technology Professionals among many others. His research interest includes Cryptography, Information Security, Computer Arithmetic and Parallel Computing.

Logunleko, K.B attended both Local and International conferences and has several (Conferences, Journals, Manuscripts) publications.



Abolore Muhamin LOGUNLEKO Ph.D. bagged BSc Mathematical Sciences (Computer Science) from the University of Agriculture, Abeokuta, Ogun State, Nigeria in the year 2007. He bagged Master of Science Computer Science from premier university, University of Ibadan, Nigeria in the year 2012 and Doctor of Philosophy Computer Science (Information Security) at Kwara State University, Malete, Kwara State, Nigeria in the year 2022. At the moment, he is a Lecturer at Department of Computer Science, Gateway ICT Polytechnic, Saapade, Ogun State, Nigeria. He

authored and co-authored many publications both in international and national journals. His research interest includes Cybersecurity, Information Security, Cryptography, Computer Arithmetic, Parallel Computing, Software Development and Database System. He belongs to many professional bodies such as the British Computer Society, Computer Professional Regulation Council of Nigeria, Nigeria Computer Society, Information Technology Security System Professional, Academia in Information Technology Professionals, Nigeria Information Tech. Professional in Civil & Public Service etc.



Ayisat Wuraola ASAJU-GBOLAGADE Ph.D. is a Lecturer I in the Department of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Ilorin, Nigeria with about nine years of university teaching, research, and administrative experiences. She served as a Students Level Adviser, Faculty Representative to Junior Staff Appointment and Promotion Committee, her area of research interest cut across Data Science, Machine Learning, Artificial Intelligence and Residue Number systems. She has to her credit over twenty publications in reputable outlets like Elsevier, Springer, among others covering journals, edited conference proceedings and chapters in books. She was part of the team that won 2020 ETF Institutional research grant and 2021 National TETFUND grant.



Akinbowale Nathaniel BABATUNDE received B.Sc, M.Sc and Ph.D. degrees from the Department of Computer Science, University of Ilorin. He joined Kwara State University, Malete as a Lecturer in the Department of Computer Science, in 2013. He has authored and co-authored several publications from both international and national journals. He is a member of Computer Professionals of Nigeria, Institute of Electrical and Electronics Engineers (IEEE) and of the Internet Society. His research interest includes information security, computer arithmetic and natural language processing.



Kazeem Alagbe GBOLAGADE received his Ph.D. in Computer Science from the Delft University of Technology in the Netherlands, his M.Sc and BSc in Computer Science from the University of Ibadan and Ilorin, respectively, in Nigeria. He began his lecturing career at the Olabisi Onabanjo University, Ago-Iwoye, Ogun State, Nigeria in January 2002 and rose through the ranks to be appointed a professor of Computer Science in March, 2014 at the prestigious Kwara State University, Malete, Kwara State, Nigeria. Professor Gbolagade has supervised twenty-two Ph.D. Students to completion and a number of Students at MSc and undergraduate levels. He is the author of more than 140 scholarly works published in prestigious peer-reviewed journals and conferences. In June 2018, he was given the inaugural Post Professorial Achievement Award. He is a FELLOW of Pan African Institute for Entrepreneurship and Community Development (FPAI), as well as a fellow of Nigeria Computer Society (FNCS). Among many other professional associations, Professor Gbolagade is an active member of the Computer Professionals (Registration Council of Nigeria)-CPN.



This is an open access article distributed under the terms and conditions of the Creative Commons Attribution-NonCommercial 4.0 International License.