International Journal of Basic and Applied Science 10 (2) (2021) 42-50

Published by: IOCSCIENCE

International Journal of Basic and Applied Science

Journal homepage: www.ijobas.pelnus.ac.id

Improve The Security of The Vigenère Cypher Algorithm by Modifying the Encoding Table and Key

Agung Purnomo Sidik
Department of Science & Technology, Universitas Pembangunan Panca Budi, Indonesia
agung@dosen.pancabudi.ac.id

Article Info

Article history:

Received Jun 19, 2021
Revised Jul 17, 2021
Accepted Sep 23, 2021

Keywords:

Vigenère Cipher;
RSA;
Blum-Blum Shub;
Encoding Table.

ABSTRACT

This investigation was conducted to move forward the security of the vigenère cypher calculation by tending to the key conveyance issue and the most key issue within the calculation if cryptoanalysts effectively got it. The working demonstrates employments a 512-bit Electronic Code Book (ECB). The essential key utilized was a 512-bit arbitrary key. The randomized encoding table was also generated by the Blum Blum Shub calculation employing a scrambler key. The RSA calculation was utilized to scramble 512-bits random keys and scrambler keys. It comes about to appear that the proposed calculation does not include a key dissemination issue. With a randomized encoding table, the 512-bits arbitrary key cannot be utilized to unscramble the ciphertext off chance that cryptanalysis did not know the mixed table utilized. The preparing time appeared the proportion of the contrast of the vigenère cypher calculation. The proposed calculation was not as well essentially distinctive, and the longer the message was handled, the proportion of the time distinction possessed was moreover getting closer to one, which implies the distinction was not critical. The coming about ciphertext was too exceptionally secure, where the coming about ciphertext was irregular since the key utilized was random, and there were 3.75×10^{126} keys combination. This algorithm can be an alternative algorithm that is fast, safe, and easy to implement.

Corresponding Author:

Agung Purnomo Sidik,
Department of Science & Technology,
Universitas Pembangunan Panca Budi, Indonesia,
Jl. Gatot Subroto No.km, Simpang Tj., Kec. Medan Sunggal, Kota Medan, Sumatera Utara 20122
Email: agung@dosen.pancabudi.ac.id

1. INTRODUCTION

Vigenère cypher is an algorithm designed for a long time and is included in the class of classic algorithms because it operates on a character scale instead of bits and a simple encryption and decryption calculation process[1][2]. Nevertheless, the vigenère cypher can be categorized as a modern cryptographic algorithm if the encryption and decryption process is carried out using the XOR technique to operate on a bit scale[3][4]. Although simple, this algorithm is quite safe, and because of its simplicity, the Vigenère cypher algorithm is an algorithm that is very light in terms of

algorithm complexity, so this algorithm only requires a small number of computer resources and time for the encryption and decryption process [5].

In the encryption and decryption process, the algorithm vigenère cypher requires an encoding table that will encode every character of plain text and ciphertext, and based on this table, the encryption and decryption process is carried out with a predetermined key[6][7]. One of the disadvantages of the vigenère cypher algorithm is that the encoding table is easy to guess[8][9][10]. The encoding table is arranged sequentially so that eavesdroppers will easily know or guess the encoding table's arrangement[11][12]. Therefore, if the encryption and decryption keys fall into the hands of the eavesdropper, the eavesdropper can be sure of decrypting the ciphertext back into plain text[13][14]. For this reason, this study will investigate how to scramble the encoding table with completely randomized randomization so that if the key falls into the hands of eavesdroppers, the confidentiality of the message can still be maintained [15][16].

Vigenère cypher is also one of the symmetric algorithms with the same encryption and decryption key so that it is like an algorithm another symmetric, one of the other weaknesses of the Vigenère cypher algorithm is the problem of key distribution[17]. It means that the vigenère cypher has a problem with how to send the decryption encryption key completely securely. There is no truly secure way to distribute the key from a sender to receiver so that recipients can decrypt the ciphertext [18].

In this study, the operating model use the Electronic Code Book (ECB) block cypher operating mode of 512-bits or 64 characters long. The primary key used in the vigenère cypher is a random key of 512-bits or 64 characters long[19][20]. The key is generated with the Blum Blum Shub algorithm to overcome the primary key obtained by the cryptanalyst. A randomized encoding table is used that is generated by the Blum Blum Shub algorithm using a random key to solve the key distribution problem[21][22][23]. The RSA public key algorithm is used to encrypt a 512-bit random key and a scrambler key so that the properties of the two keys are no longer secret[24][25][26].

2. RESEARCH METHOD

The proposed algorithm scheme is divided into two areas, namely the sender's processing area and the receiver's processing area.

2.1 Message sender processing area

a. Random number generation
   The first step is to generate a random number with a random key chosen by the message's sender. This process uses the Blum Blum Shub algorithm. The result of this process will generate a series of random numbers[21][22][23].

b. Random encoding table formation
   In this process, the obtained series of random numbers is used to generate a random encoding table. The process of generating a random encoding table uses a special algorithm that has been designed as follows:
   a) Define the length of the encoding table, i.e. n; //where in this study, the length of the table is 95 characters
   b) The form of the array data structure is encoding [0..n-1];
   c) Determine the starting point i=n-1;
   d) Define the sequence of characters from start to finish that will be arranged in the encoding table in an array data structure, namely character[0..n-1];
   e) Determine the starting point q = n;
   f) Determine the starting point i = 0;
   g) Generate a random number j with the Blum-Blum Shub Algorithm and then modulo with q;
   h) Take character[j] and store it in encoding[i];
   i) Add 1 value of i, namely i++;
   j) Move one index of all index characters j+1 to n-1 in the character array to the left;

k) At least 1 value of q, namely q--;
l) Repeat steps 7 until q=1;

c. 512-bits symmetric key generation (Primary Key)
The next step is to generate a 512-bit random symmetric key using the Blum Blum Shub algorithm. This symmetric key is used as the primary key to encrypt plain text into cypher text.

d. Encrypting plain text into cypher text
At this stage, the encryption process is carried out from plain text to cypher text. A generated 512-bit symmetric key is required, and a generated random encoding table.

e. Encryption of the primary key into a cypher key
This stage is done by encrypting the symmetric key using the RSA algorithm, and the public key becomes the cypher key.

f. Encryption of the scrambler key into a cypher key scramble
The scrambler key is encrypted using the public key obtained from the recipient of the message. The result of the encryption key scrambler becomes a cypher key scrambler.

The message sending processing area's output produces three main outputs: cypher text, cypher key, and cypher key scrambler. The third data is sent to the recipient of the message. Free shipping is carried out on safe or unsecured routes because the three data are no longer confidential (free).

2.2 Message recipient processing area

a. Generating the private key and public key[6].
The first step that the recipient does is to generate a private key and a public key using two prime numbers p and q. The resulting public key is sent to the sender of the message. This public is not confidential and can be sent from insecure channels. While the private key generated is still stored by the recipient of the message and may not be distributed or known by other parties.

b. Scrambler cypher key decryption
The message recipient's random cypher key is decrypted using the RSA algorithm with a private key. The result of this decryption generates a random key which is used for the random number generation process[26][27].

c. Random number generation
Generating random numbers is done by providing a random key decrypted from the cypher key scrambler. This process generates a series of random numbers, which are used to generate a random encoding table.

d. Random encoding table formation
The series of random numbers generated in the previous process becomes an input to generate a random encoding table used for the decryption process.

e. Cipher key description
The cypher key received by the recipient of the message is then decrypted using RSA algorithm with a private key that the recipient of the message still stores. Results decryption generates a 512-bit symmetric key (primary key)[27].

f. Cipher text description
After the random encoding table is generated successfully and the 512-bits symmetric key (primary key) has been obtained from the decryption of the cypher key, the cypher text can be decrypted to produce plain text. The decryption process is carried out using the Vigenère Cipher algorithm with XOR technique in the 512-bits Electronic Code Book (ECB) operating mode.

The Method can be viewed as figure below:

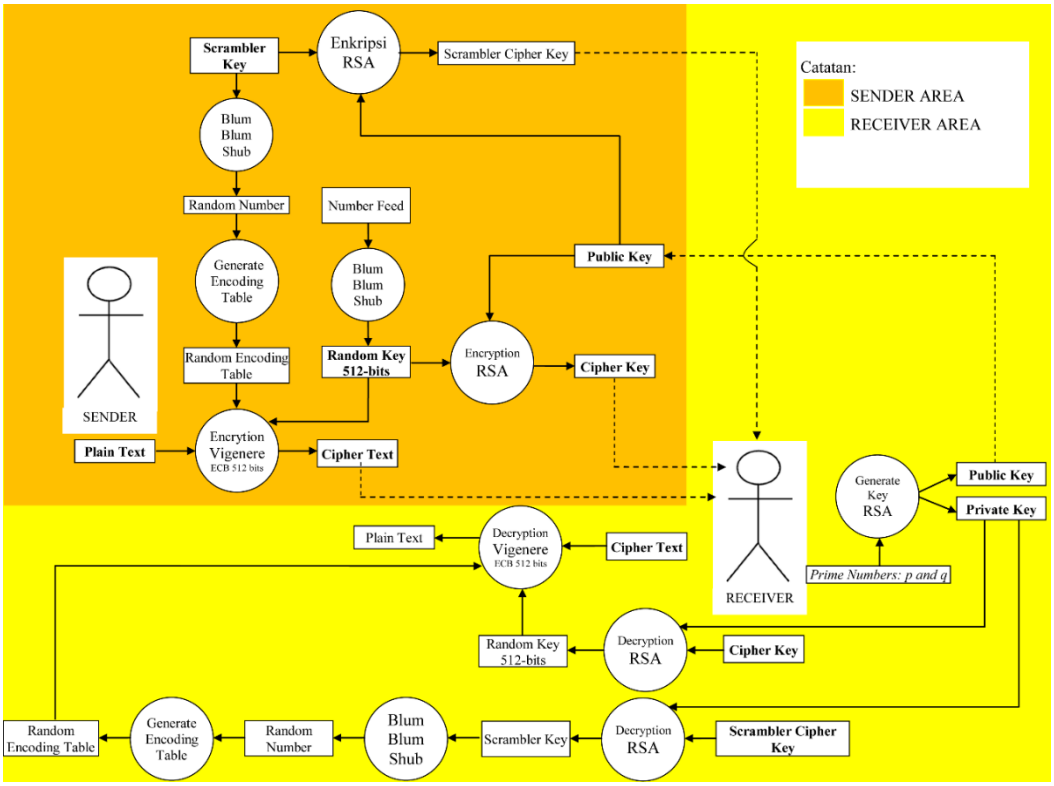

Figure 1. Process Flow Used in Research

3. RESULTS AND DISCUSSIONS

Private and public key generation result:

Private key (d,n) : 6cd0f,18a9ab
Public key (e,n) : 4cf,18a9ab

Random key generation result:

Scrambler key (s,n) : 69b63,8cdfd
Scrambler cypher key (s,n) : 4e815,d9d31

The results of randomizing the encoding table with the scrambler key can be seen in the following table:

Table 1. Comparison of Encoding Tables Before and After Randomization

Before Randomized (ASCII)			After Randomized (ASCII)		
Index	Decimal	Character	Index	Decimal	Character
0	32	<spasi>	0	39	'
1	33	!	1	55	7
2	34	"	2	103	g
3	35	#	3	58	:
4	36	$	4	88	X
5	37	%	5	90	Z
6	38	&	6	35	#
7	39	'	7	61	=
8	40	(	8	65	A
9	41	)	9	86	V
10	42	*	10	121	y
11	43	+	11	102	f
12	44	,	12	48	0

Before Randomized (ASCII)			After Randomized (ASCII)		
Index	Decimal	Character	Index	Decimal	Character
13	45	-	13	117	u
14	46	.	14	34	"
15	47	/	15	38	&
16	48	0	16	92	\
17	49	1	17	94	^
18	50	2	18	64	@
19	51	3	19	87	W
20	52	4	20	73	I
21	53	5	21	111	o
22	54	6	22	112	p
23	55	7	23	37	%
24	56	8	24	108	l
25	57	9	25	123	{
26	58	:	26	43	+
27	59	;	27	126	~
28	60	<	28	119	w
29	61	=	29	122	z
30	62	>	30	96	`
31	63	?	31	76	L
32	64	@	32	109	m
33	65	A	33	47	/
34	66	B	34	97	a
35	67	C	35	104	h
36	68	D	36	105	i
37	69	E	37	95	_
38	70	F	38	83	S
39	71	G	39	36	$
40	72	H	40	81	Q
41	73	I	41	82	R
42	74	J	42	41	)
43	75	K	43	79	O
44	76	L	44	98	b
45	77	M	45	99	c
46	78	N	46	63	?
47	79	O	47	52	4
48	80	P	48	113	q
49	81	Q	49	118	v
50	82	R	50	89	Y
51	83	S	51	51	3
52	84	T	52	80	P
53	85	U	53	59	;
54	86	V	54	70	F
55	87	W	55	74	J
56	88	X	56	42	*
57	89	Y	57	110	n
58	90	Z	58	33	!
59	91	[	59	120	x
60	92	\	60	77	M
61	93	]	61	84	T
62	94	^	62	62	>
63	95	_	63	69	E
64	96	`	64	124	|
65	97	a	65	50	2
66	98	b	66	116	t
67	99	c	67	40	(

Before Randomized (ASCII)			After Randomized (ASCII)		
Index	Decimal	Character	Index	Decimal	Character
68	100	d	68	106	j
69	101	e	69	49	1
70	102	f	70	107	k
71	103	g	71	78	N
72	104	h	72	46	.
73	105	i	73	114	r
74	106	j	74	67	C
75	107	k	75	60	<
76	108	l	76	57	9
77	109	m	77	72	H
78	110	n	78	44	,
79	111	o	79	100	d
80	112	p	80	101	e
81	113	q	81	54	6
82	114	r	82	115	s
83	115	s	83	32	<spasi>
84	116	t	84	71	G
85	117	u	85	66	B
86	118	v	86	85	U
87	119	w	87	93	]
88	120	x	88	53	5
89	121	y	89	45	-
90	122	z	90	68	D
91	123	{	91	91	[
92	124	|	92	75	K
93	125	}	93	125	}
94	126	~	94	56	8

The scrambled encoding table can no longer be guessed in order, making encryption much more secure. Plain text is encrypted using an encoding table that has been scrambled with a random key. Then the random key and the random key are encrypted with the RSA algorithm using a public key.

Plaintext : Soekarno proclaimed Indonesia's independence on August 17, 1945 in Jakarta

512-bit Random Key:

w"A&B{aaS<f?L^,ZynyZSJT%kyJDQu,P)DVnfpMJ-O^JRaoLxi1U~J1SK1c4s6h$

Ciphertext :

th5Bl:[J+gG(9R^R))-5!^cbmD)LCu2Q,P5)/=p$CZ>QzJ1W(vNX"+no}P/oE/w~|N[kl'joV"

Cipher Keys:

dd868.180c64.16b616.bfd64.76c46.8d595.cef2e.cef2e.f1fa8.136514.177c8f.d83fd.3225.c44b.1c419.17e7f7.bfae8.c0a16.bfae8.17e7f7.f1fa8.170c9c.c3ffd.99205.698fb.bfae8.170c9c.144473.129253.938e8.1c419.117c8.fae76.144473.37eee.c0a16.177c8f.152f70.2907d.170c9c.170add.92980.c44b.170c9c.b9269.cef2e.70ed2.3225.ae964.29ad0.5af11.5b42f.180d93.170c9c.5af11.f1fa8.c8c4d.5af11.3e6cf.a482.13aa80.7fa6.181358.15ba4e.

Comparison of the encryption and decryption process of the pure Vigenère Cipher algorithm with the designed algorithm can be viewed as the table below: (The unit of time used is in seconds)

Table 2. Results of Comparison of Processing Time of the Vigenère Cipher Algorithm with the designed algorithm

Message Length	**Pure Vigenère *Cipher***		**Designed Algorithm**	
(Character)	Encryption Time	Decryption Time	Encryption Time	Decryption Time
10	0,001148	0,000164	0,005161	0,00277686
100	0,002538	0,001528	0,006763	0,00415897
1000	0,0098989	0,0079739	0,0175369	0,01823793
10000	0,0561759	0,0444061	0,095085	0,09027802
100000	0,6624131	0,5518869	0,7852581	0,74352112
500000	3,0910089	3,0337591	3,587478	3,39756885
1000000	8,143666	7,7332649	8,6350069	7,85362291

The comparison results can also be seen in the following graph::

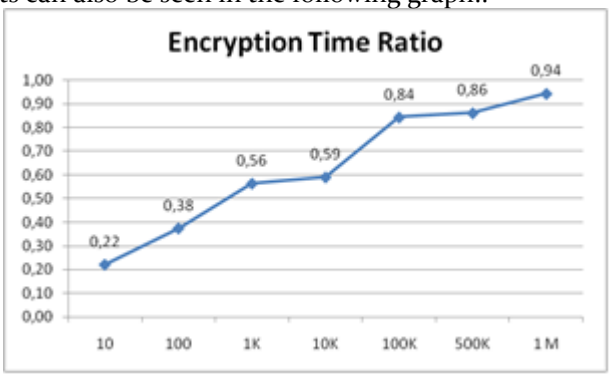

Figure 2. Encryption Time Ratio Comparison Graph

The graph shows that the larger the encrypted plain text, the closer the encryption time ratio is to one. It means that the difference in encryption time using the Vigenre Cipher algorithm before being modified with the Vigenère Cipher algorithm after being modified is not much different, especially if the data is large enough.

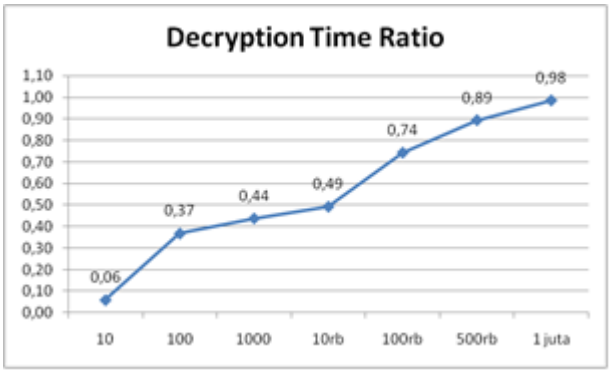

Figure 3. Decryption Time Ratio Comparison Graph

The graph shows that the larger the cypher text that is decrypted, the comparison of the decryption time is also getting closer to one. It means that the difference in decryption time using the Vigenre Cipher algorithm before being modified with the Vigenère Cipher algorithm after being modified is not much different, especially if the data is large enough.

Plain text is encrypted with a key of 64 characters or 512-bits so that the plain text will produce 95^{64} or 3.75 x 10^{126} variations of cypher text. The number 95 was obtained because this study used 95 ASCII characters in the encoding table. Key combinations that occur are 95^{64} or 3.75 x 10^{126}.

So that cryptanalysts with exhaustive search techniques or try all possible keys one by one takes a very long time so that this algorithm is safe from attacks with exhaustive search techniques.

The key is generated randomly with the Blum Blum Shub algorithm so that the resulting ciphertext will be completely random. There is no other way that can be done by the cryptanalyst other than trying one by one the possible keys to find the right key. It will take a very long time and is very tiring.

4. CONCLUSION

Using a randomized encoding table, if the cryptanalyst succeeds in only getting the primary key and cypher text, then the decryption process cannot be carried out. The master key that is owned becomes useless because the cryptanalyst requires the same encoding table when the encryption process uses to perform decryption. A random key is needed to generate the same randomized encoding table. The RSA algorithm also allows the primary key and random key that has a secret nature to change to be no longer secret. It can be concluded that the modified design of the vigenre cypher algorithm in this study can overcome the problem of key distribution and the problem of the primary key. It was successfully obtained by cryptanalysts so that the design of this algorithm is much more robust and more secure than the usual vigenre cypher algorithm.

REFERENCES

[1] B. Delman, "Genetic algorithms in cryptography," 2004.

[2] J. P. Sermeno, K. A. S. Secugal, and N. E. Mistio, "Modified Vigenere cryptosystem: An integrated data encryption module for learning management system," *Int. J. Appl. Sci. Eng.*, vol. 18, no. 4, pp. 1–10, 2021.

[3] M. Abd Zaid and S. Hassan, "Survey on Modern Cryptography," *J. Kufa Math. Comput.*, vol. 7, no. 1, pp. 1–8, 2021.

[4] A. Malik, S. Gupta, and S. Dhall, "Analysis of traditional and modern image encryption algorithms under realistic ambience," *Multimed. Tools Appl.*, vol. 79, no. 37, pp. 27941–27993, 2020.

[5] Q.-A. Kester, "A cryptosystem based on Vigenère cipher with varying key," *Int. J. Adv. Res. Comput. Eng. Technol.*, vol. 1, no. 10, pp. 108–113, 2012.

[6] G. Ye, K. Jiao, H. Wu, C. Pan, and X. Huang, "An asymmetric image encryption algorithm based on a fractional-order chaotic system and the RSA public-key cryptosystem," *Int. J. Bifurc. Chaos*, vol. 30, no. 15, p. 2050233, 2020.

[7] K. Kim, F. A. Alfouzan, and H. Kim, "Cyber-Attack Scoring Model Based on the Offensive Cybersecurity Framework," *Appl. Sci.*, vol. 11, no. 16, p. 7738, 2021.

[8] V. K. Mittal and M. Mukhija, "Cryptosystem Based on Modified Vigenere Cipher using Encryption Technique," 2019.

[9] T. M. Aung and N. N. Hla, "A Complex Polyalphabetic Cipher Technique Myanmar Polyalphabetic Cipher," in *2019 International Conference on Computer Communication and Informatics (ICCCI)*, 2019, pp. 1–9.

[10] M. H. Purwidiantoro and D. F. K. S. Wibowo, "Super Encryption Concepts using Vigenere Cipher Modification to Produce Color Imaginary as Ciphertext," 2020.

[11] H.-H. Li, L.-H. Gong, and N.-R. Zhou, "New semi-quantum key agreement protocol based on high-dimensional single-particle states," *Chinese Phys. B*, vol. 29, no. 11, p. 110304, 2020.

[12] J. Choi, J. Yu, S. Hyun, and H. Kim, "Digital forensic analysis of encrypted database files in instant messaging applications on Windows operating systems: Case study with KakaoTalk, NateOn and QQ messenger," *Digit. Investig.*, vol. 28, pp. S50–S59, 2019.

[13] D. Rupprecht, K. Kohls, T. Holz, and C. Pöpper, "Call Me Maybe: Eavesdropping Encrypted {LTE} Calls With ReVoLTE," in *29th {USENIX} Security Symposium ({USENIX} Security 20)*, 2020, pp. 73–88.

[14] C. Wang, A. Wang, J. Xu, Q. Wang, and F. Zhou, "Outsourced privacy-preserving decision tree classification service over encrypted data," *J. Inf. Secur. Appl.*, vol. 53, p. 102517, 2020.

[15] A. A. Soofi, I. Riaz, and U. Rasheed, "An enhanced Vigenere cipher for data security," *Int. J. Sci. Technol. Res*, vol. 5, no. 3, pp. 141–145, 2016.

[16] M. A. H. Al-Halboosi, "Agile Encryption Scheme for Multimedia Files Using Random Data." Middle East University, 2021.

[17] B. Triandi, E. Ekadiansyah, R. Puspasari, L. T. Iwan, and F. Rahmad, "Improve Security Algorithm Cryptography Vigenere Cipher Using Chaos Functions," in *2018 6th International Conference on Cyber*

and IT Service Management (CITSM), 2018, pp. 1–5.
[18] M. Thangavel, P. Varalakshmi, M. Murrali, and K. Nithya, "An enhanced and secured RSA key generation scheme (ESRKGS)," *J. Inf. Secur. Appl.*, vol. 20, pp. 3–10, 2015.
[19] S. Chandra, "Peningkatan Keamanan Algoritma Vigenère Cipher dengan Teknik Pengacakan Tabel Encoding," 2019.
[20] C. Cid, S. Murphy, and V. Rijmen, "ANALYSIS OF LIGHTWEIGHT AND EFFICIENT SYMMETRIC-KEY PRIMITIVES."
[21] Y. D. Vybornova, "Password-based key derivation function as one of Blum-Blum-Shub pseudo-random generator applications," *Procedia Eng.*, vol. 201, pp. 428–435, 2017.
[22] C. D. Omorog, B. D. Gerardo, and R. P. Medina, "Enhanced pseudorandom number generator based on Blum-Blum-Shub and elliptic curves," in *2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, 2018, pp. 269–274.
[23] E. Koopahi and S. E. Borujeni, "Secure scan-based design using Blum Blum Shub algorithm," in *2016 IEEE East-West Design & Test Symposium (EWDTS)*, 2016, pp. 1–5.
[24] P. Patil, P. Narayankar, D. G. Narayan, and S. M. Meena, "A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish," *Procedia Comput. Sci.*, vol. 78, pp. 617–624, 2016.
[25] L. Gong, K. Qiu, C. Deng, and N. Zhou, "An optical image compression and encryption scheme based on compressive sensing and RSA algorithm," *Opt. Lasers Eng.*, vol. 121, pp. 169–180, 2019.
[26] R. Fotohi, S. Firoozi Bari, and M. Yusefi, "Securing wireless sensor networks against denial-of-sleep attacks using RSA cryptography algorithm and interlock protocol," *Int. J. Commun. Syst.*, vol. 33, no. 4, p. e4234, 2020.
[27] C. Hazay, G. L. Mikkelsen, T. Rabin, T. Toft, and A. A. Nicolosi, "Efficient RSA key generation and threshold paillier in the two-party setting," *J. Cryptol.*, vol. 32, no. 2, pp. 265–323, 2019.