RESEARCH ARTICLE OPEN ACCESS

KASUMI Block Cipher's Use of Chaos-Based Cryptography

MAHESH TUBAKI
RESEARCH SCHOLAR, srinivas university Mangaluru

Dr. Rajanna G. S
Professor in E&C engineering
Srinivas University Mangaluru

ABSTRACT
The hardest thing to do in network communication is send data securely. A technique for maintaining message or data secrecy, or safe data transfer, is cryptography. This cutting-edge technology is crucial for network security. The field of cellular networks is the most significant one for communication. Every day, a great deal of research is being done on cryptography, which is still in its early stages of development. Therefore, a significant amount of research methodology or work remains necessary for secure communication in cellular networks. This study proposes a novel cryptosystem based on the KASUMI block cipher and the Lorenz equations of chaos theory. Random numbers are important in many domains, including cryptography. In our proposed study, Lorenz equations are used to create random numbers based on chaos theory. The NIST test suite is used to verify the unpredictability of generated random numbers. The KASUMI block cipher is taken into consideration for data encryption. Three parameters—balanced output, Hamming distance, and avalanche effect—are used to examine the encrypted data performance evaluation that the suggested approach produces. An experimental finding provides optimal randomness, leading to enhanced cryptosystem performance.

Keywords: *Chaos Theory, Cryptography, KASUMI Lorenz Equation*

I. INTRODUCTION

To address the needs of real-time, secure information transmission via networks, communication security is becoming increasingly important. All our everyday interactions and transactions in the current world take place across mobile network infrastructures. Since the previous ten years, there has been a more exponential rise in the use of mobile services. In the sphere of network communication, maintaining data security is a crucial and difficult undertaking. Network security methods come in a plethora of forms, one of which is new technology: cryptography. Cryptography will offer a great deal of control over authentication and privacy *[3]*. A mobile network user's primary privacy needs were in the areas of voice, call setup data, user location, user identification, and calling patterns.

Encrypting data before sending it to a recipient or permitted network is crucial for maintaining privacy in network communication and cellular networks. Random numbers are essential to the encryption process as a key. Random Number Generators (RNGs) are the fundamental building blocks of cryptography and are used in a wide range of cryptographic applications, including the creation of secret keys, protocol authentication, and attack defense *[2]*. The progress of research on cryptography and chaos-based random number generation is currently ongoing, and successful research is still required for secure communication on cellular networks.

Numerous fields, such as wireless communications, sensor networks, and military equipment, can benefit from the suggested effort. FPGAs are the most extensively used platforms for implementing cryptographic circuits because of their potential for high-end flexibility, complex on-chip interconnects, and programmable computing at run-time. Furthermore, the run-time reconfiguration of FPGAs using DPR and RLUT (SRL32 primitive) has undoubtedly contributed to the development of a secure design package. By altering the pre-defined

LUT contents internally, the RLUT reconfigures the hardware core. However, the DPR technique does not need arbitration of the active FPGA operation, allowing a portion of the bit stream to be uploaded dynamically at predetermined stages / set-timings.

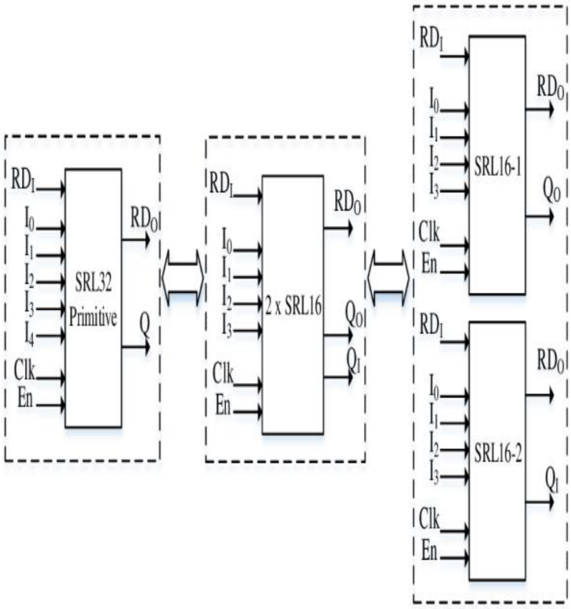


Figure No. 1: RLUT (configured as 1 × SRL32 primitive or 2 × SRL16).

Thus, the need for secure communication on cellular networks led us to create a brand-new cryptosystem based on random generators. Lorenz equations are used in the chaos theory to generate random numbers. Simulation is carried out using the obtained random numbers for data encryption. Here the encryption is done using the KASUMI method. An improved version of the MISTY1 encryption, the KASUMI block cipher is also chosen as the foundation for the 3GPP integrity and confidentiality approach. The remainder of the document is structured as follows: The many methods that have been examined as part of the literature review are explained in Section II. The experimental findings and methods for the proposed system are presented in Sections III and IV, respectively. Finally, we included a reference in Section V after we wrapped up our work.

II. LITERATURE SURVEY

A cryptographic method for safe communication in wireless networks was introduced by ***Fagen Li et al. [1]***. For anonymity, they employed identity-based encryption in this case. For secure broadcasting between a sensor node and an internet host, this method was developed to be both offline and online heterogeneous in terms of signcryption. A network-supported direct broadcasting security solution for faulty cellular connectivity was provided by ***Alexander Ometov et al. [2].*** With this method, the state of the art was reviewed, and a new algorithm was created to maintain the security features of neighboring devices in the event of cellular connectivity failure.

A random bit generator that complies with FIPS 140-3 and is based on SRAM PUF was introduced by ***Vincent van der Leest et.al [3].*** Large measurement sets are taken into consideration while determining the noise minimum entropy in SRAM patterns. The compression factor for the condition method is provided by the computed min entropy. The 256-bit true random seed is removed from the memory using this conditioning procedure.

True random number generation via discrete period chaos was proposed by ***Ihsan Cicek et al. [4]***. This method makes use of the Skew Tent Map approach. Here, the randomness is computed using a mathematical model of a genuine random number generator. Chaos-theory-based cloud computing cryptography was introduced by ***Paul Tobin et al. [5]***. An OTP-based encryption method is described in the system in question. Here, chaotic cryptography encryptor circuits are constructed via simulation. A method for creating pseudorandom numbers was given by ***Ismail Ozturk et al. [6]***. Using a Lu-like chaotic system, this approach can display both Chen-like and Lorenz-like chaotic technique properties for a range of factor values.

Using the KASUMI cipher approach, ***Wentan Yi et al. [7]*** created a zero-correlation multidimensional cryptanalysis. This technique creates a linear model with a 5-round zero correlation. On KASUMI, the zero-correlation attack with six rounds is showcased. Several block ciphers were examined in terms of functionality and performance by ***Thomas Eisenbarth et al. [8]***. The block ciphers KASUMI, KLEIN, AES, HIGHT, KATAN, HIGHT, TEA, mCrypton, PRESENT, SEA, and NOEKEON are examined in this study that has been cited. The effectiveness of these block ciphers is assessed after implementation. A modified F8 encryption method based on the KASUMI block cipher was created by ***Nabil H. Shaker et al.*** **[9]**. The updated modified edition of the KASUMI block cipher is given in this linked publication boxes in

KASUMI are the focus of this customization, and S boxes undergo testing to confirm cryptographic functionality.

The suggested RLUT and DPR based designs are presented in the study after a brief overview of reconfiguration approaches and a discussion of the MISTY1 and KASUMI algorithms. Lastly, a summary and conclusion are provided for the results.

III. METHODOLOGY

The process used in the suggested technique is described in this section. Here, we introduced a novel cryptosystem that combines the KASUMI block cipher with random numbers based on Lorenz equations. This cryptosystem is made up of an encryption module and a Lorenz generating key module that were created using chaotic Lorenz equations. Using the chaos-based Lorenz equation, random numbers are initially created in the first stage. The Lorenz generating key for the encryption module is then created using the produced random numbers. Secure encryption requires the creation of S-Boxes, which are made possible by this Lorenz generating key [8]. Data may be found in cellular communication in the following formats: text, image, audio, and video messages. Here, input is defined as plain text data. Subsequently, a subset of plain data is sent to an encryption module for cyphering. Between plain data, 128-bit subkeys, and random numbers based on Lorenz, encryption is carried out. The data encryption in this encryption module uses the KASUMI cipher. Fig. 2 shows the proposed system's operating process visually.

3.1: Lorenz Equation

In cryptography, random numbers are crucial for data security. Random numbers are generated in the proposed system using Lorenz equations based on chaos. One Lorenz equation was proposed in 1963 by an individual going by the name of E.N. Lorenz. Different chaotic systems, such as logistic maps, can display peculiar attractors without taking dimensionality into account. Three differential equations combine to form the Lorenz attracter. The following are the Lorenz equations (1), (2), and (3):

$$\frac{dx}{dt} = \sigma y - \sigma x \qquad (1)$$

$$\frac{dy}{dt} = \varrho x - xz - y \qquad (2)$$

$$\frac{dz}{dt} = xy - \beta z \qquad (3)$$

The system parameters are where, h, and indicate the time, and where, and generates the system state. Nonlinear equations are typically exceedingly difficult to solve analytically, necessitating the use of numerical techniques. Using Euler's approach, the simplest version of these equations is presented in the proposed system. (4), (5), and (6) refer to the modified Euler's equation for the Lorenz equations:

$$Xn + 1 = Xn + h(aYn - aXn) \qquad (4)$$

$$Yn + 1 = Yn + h(bXn - XnZn - Yn) \qquad (5)$$

$$Zn + 1 = Zn + h(XnYn - cZn) \qquad (6)$$

The constant terms are where, here, and. The finest genuine random numbers are produced by using equations (5), (6), and (4). To build SBoxes, which are necessary for encryption, these generated random integers are utilized as the Lorenz generating key. The purpose of these S-Boxes is to stop linear structuring.

3.2: Data Encryption

Data can be transformed from one recognized form to another via encryption. It's crucial to use an encryption mechanism before transferring any data to the recipient. The optimal encryption technology selection results in optimum security and system performance. The 64-bit block cipher known as the KASUMI cipher is employed for encryption in our suggested method. Fig. 2 shows the data encryption's functional flow.

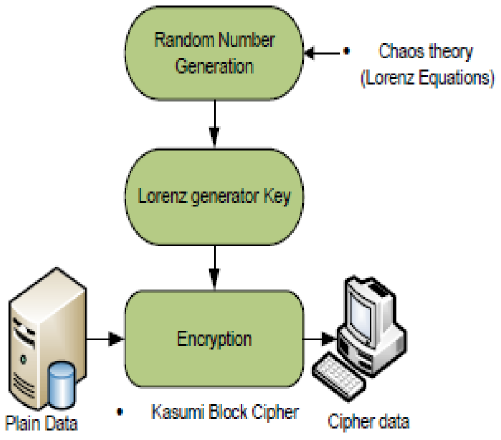


Figure No. 2: Proposed Cryptosystem

3.3: KASUMI Block Cipher

One of the widely used encryption methods in 3G mobile communication is KASUMI. KASUMI is made from an eight-round Feistel structure. It uses a 128-bit secret key and 64-bit input data. The output is 64-bit encryption. The suggested approach uses a 128-bit key for encryption on plain data, which is required for encryption. Figure 3 depicts the KASUMI technique's functional flow. Algorithm 1 outlines the procedures for the KASUMI encryption technology. The sections that follow provide explanations of the various parts of the KASUMI method.

A. Function (fi)

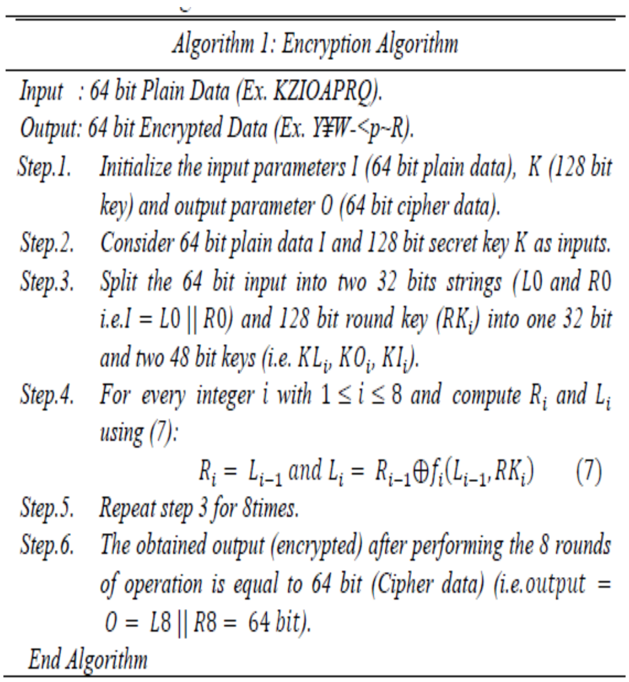
The function considers round key and outputs a 32-bit value based on a 32-bit input. The three subkeys are constructed using the round key. The function is built from two subfunctions, each of which is coupled with a subfunction. The function has two distinct formats depending on whether the round is odd or even. For odd rounds, namely 1, 3, 5, and 7, the following is explained in (8):

$$fi(I, RKi) = FO(FL(I, KLi), KOi, KIi) \qquad (8)$$

Additionally, (9) describes the even rounds, which are 2, 4, 6, and 8.

$$fi(I, RKi) = FL(FO(I, KOi, KIi), KLi) \qquad (9)$$

Data is transmitted through the FO function in even rounds and the FL function in odd rounds.



```
Algorithm 1: Encryption Algorithm
Input  : 64 bit Plain Data (Ex. KZIOAPRQ).
Output: 64 bit Encrypted Data (Ex. Y₮W-<p~R).
Step.1.  Initialize the input parameters I (64 bit plain data), K (128 bit
         key) and output parameter O (64 bit cipher data).
Step.2.  Consider 64 bit plain data I and 128 bit secret key K as inputs.
Step.3.  Split the 64 bit input into two 32 bits strings (L0 and R0
         i.e.I = L0 || R0) and 128 bit round key ($RK_i$) into one 32 bit
         and two 48 bit keys (i.e. $KL_i$, $KO_i$, $KI_i$).
Step.4.  For every integer i with $1 \le i \le 8$ and compute $R_i$ and $L_i$
         using (7):
             $R_i = L_{i-1}$ and $L_i = R_{i-1} \oplus f_i(L_{i-1}, RK_i)$     (7)
Step.5.  Repeat step 3 for 8times.
Step.6.  The obtained output (encrypted) after performing the 8 rounds
         of operation is equal to 64 bit (Cipher data) (i.e.output =
         O = L8 || R8 = 64 bit).
End Algorithm
```

B. Funtion (FL)

The 32-bit key (KLi) and 32-bit plain data I are the inputs for the function FL. This function divides the 32-bit key (KLi) into two 16-bit subkeys. Where ***KLi = KLi,1** || **KLi,2.*** I=K||R, the input data, is divided into two 16-bit data sets, L and R.(10) and (11) are used to operate on two 16-bit data sets:

$$R' = R \oplus ROL\left(L \cap KL_{i,1}\right)$$

$$L' = L \oplus ROL\left(R \cap KL_{i,2}\right)$$

The output of function FL is equal to 32 bit i.e. (L′ || R′).

C. Function (FO)

The function's inputs are two 48-bit subkeys (i.e., and) and 32-bit data. In this case, the 32-bit input data is split into two 16-bit subdatasets, and. Thus, as seen in (12), 48 bit two subkeys are split up into 16 bit three subkeys:

$$KO_i = KO_{i,1} || \, KO_{i,2} || KO_{i,3}$$

$$KI_i = KI_{i,1} || \, KI_{i,2} || KI_{i,3}$$

After that for every integer j with 1≤ j ≤ 3, we define Rj and Lj using (13) and (14)

The output of function is equal to 32 bit i.e. (Lj||Rj)

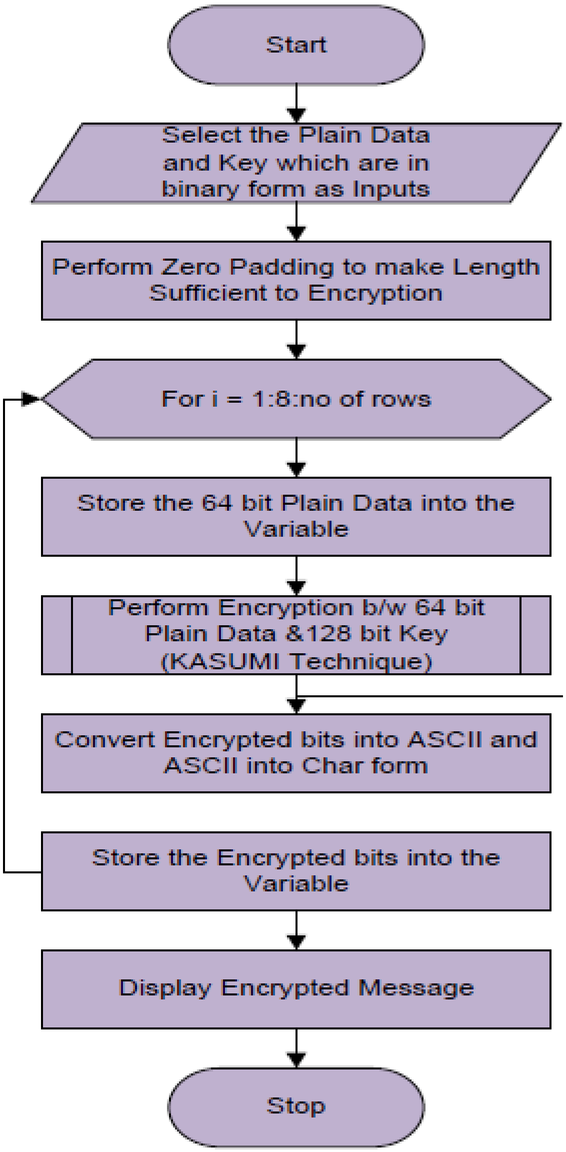

Start
Select the Plain Data and Key which are in binary form as Inputs
Perform Zero Padding to make Length Sufficient to Encryption
For i = 1:8:no of rows
Store the 64 bit Plain Data into the Variable
Perform Encryption b/w 64 bit Plain Data &128 bit Key (KASUMI Technique)
Convert Encrypted bits into ASCII and ASCII into Char form
Store the Encrypted bits into the Variable
Display Encrypted Message
Stop

Figure No. 3: Functional Flow of Data Encryption

D. Function (FI)

This function requires the usage of two S-boxes, namely and corresponds to a 7-bit input to a 7-bit output, whereas S9 corresponds to a 9-bit input to a 9-bit output. Additionally, it has two additional functions: TR (Truncate) and ZE (Zero Extend). In order to produce a 9-bit output, ZE adds two zeros to the most significant end of a 7 bit input. TR deletes the two most important bits from a 9-bit input to generate a 7-bit output. The function's functionality is carried out utilizing The output of function is equal to 16 bit i.e. (L4||R4).

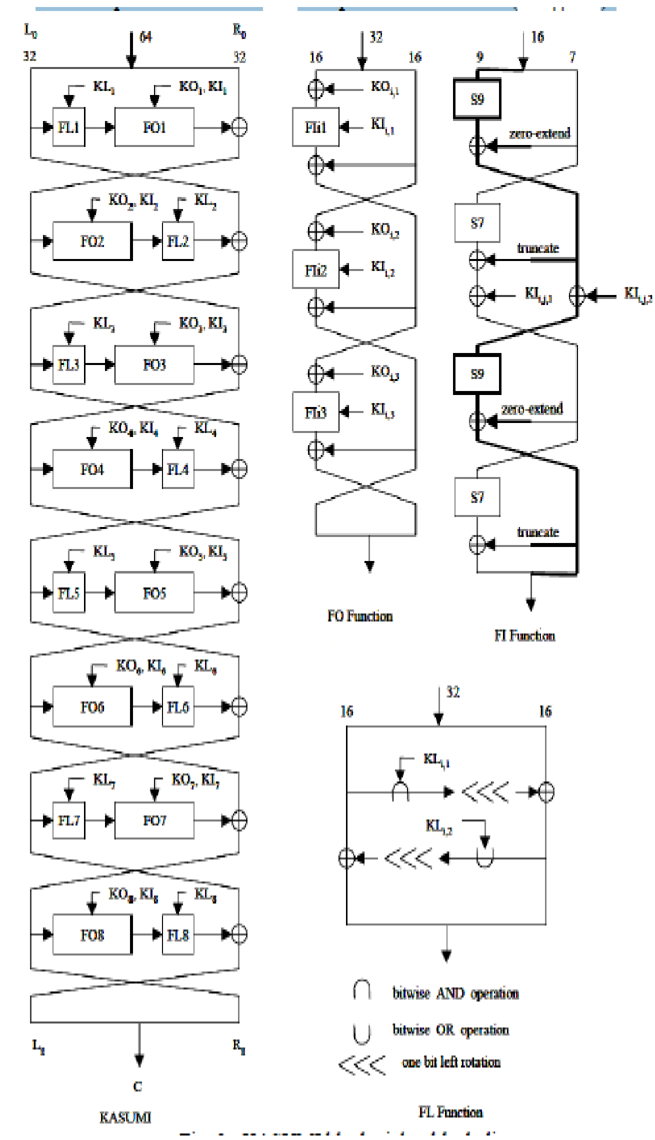


Figure No. 4: KASUMI block cipher block diagram

E. S-boxez and Key Schedule

S boxes (and) in function are made to avoid linear structure. is a permutation of 128 elements, or 0 to 127, and of 512 elements, or 0 to 511. First, random numbers are produced and used to form these S boxes. The KASUMI key is 128 bits long. 128 bits of keys, which are obtained from, are used in each KASUMI round. This 128-bit key is used to produce the subkeys needed for encryption.

IV. EXPERIMENTAL RESULTS

The findings from the assessment of the suggested system are presented in this section. Based on four security parameters—the randomness test, balanced output, Hamming distance, and avalanche effect—the suggested cryptosystem's security analysis is assessed.

4.1: Randomness Test

Using Lorenz equations based on chaos, random numbers are successfully generated in the suggested system. The NIST test suite is used to evaluate randomly generated numbers. The section below provides a brief explanation of the NIST test suite and how to acquire results from the randomness test.

A. NIST Test Suite

The statistical test suite from the National Institute of Standards and Technology (NIST) is used to assess randomness. The test consists of fifteen factors in total [10]. The produced genuine random numbers in the proposed system are subjected to a statistical test suite, NIST SP-800 22, to ensure their unpredictability. In the suggested study, eight tests are taken into consideration for this statistical test. Table I displays the findings that were received from the NIST test suite. The eight randomness tests are all passed by the suggested random numbers. It provides the best randomness results and demonstrates that the produced random numbers are in random order.

B. Balanced Output

The encrypted bits in a balanced output have an equal amount of ones and zeros. The optimal system performance for encrypted data is achieved when the proportion of 1s and 0s in the encrypted bits is about equal.

Example-1: Plain Data = RJ45JACK
=0101001001001010001101000011010101001010
010000010100001101001011,

Cipher Data = qÕÚQCÅTÔ =
011100011101010111011010001000101000011110
001010101010011010100

There are 31 1s and 33 0s in this cipher data, and these 0s and 1s are almost equal. It demonstrates that the encrypted data has the highest quality. Table II displays the balanced output of the suggested system, and Table III shows that, out of eight examples, satisfactory balanced output is produced.

C. Hamming Distance

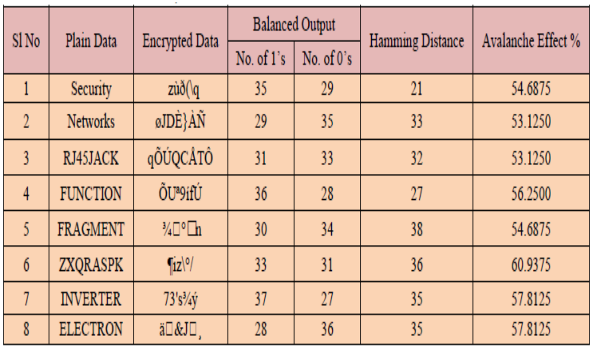
The unit of measurement for the separation of two strings of equal length is the hamming distance. It indicates how many places there are differences between the corresponding symbols [11]. In Example-1, the hamming distance between the plain and encryption data is 32. This indicates that, out of 64 bits, the bits in 32 locations are different. Greater hammer distance indicates optimal security. Over 50% of the location bits in this are different. Thus, the measured hamming distance is accurate. According to Table II, the suggested system's hamming distance yields the best outcomes in six out of eight scenarios.

Statistical Test	**P-Value**	**Result**
Frequency	289	Passed
Block Frequency (m=128)	0.04419	Passed
Cumulative Sums - Forward	0.202009	Passed
Cumulative Sums - Reverse	0.219194	Passed
Longest Runs of Ones	0.024582	Passed
Non-Overlapping Templates (m=9, B=0001)	0.328478	Passed
Linear Complexity (M=500)	0.609093	Passed

Table No. 1: Results of the NIST Statistical Randomness Tests

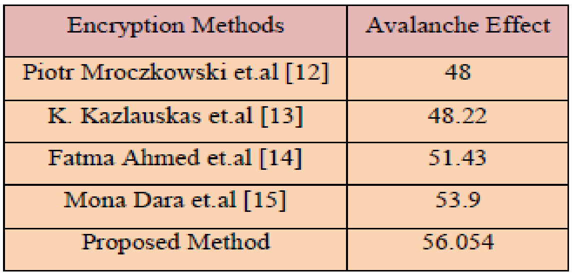
D. Strict Avalanche Criterion (SAC)

One important aspect of the cryptosystem is SAC. A single bit change in the plain data must result in completely new cipher data that differs from the cipher data that was previously created. Table II displays the Avalanche effect derived from the suggested system. In Table III, the obtained avalanche effect is compared with current techniques. The suggested system yielded an average avalanche impact of 56.054. One-bit changes in plain data, according to SAC, results in more than 50% changes in the ciphered content. In contrast to current techniques [12] [13] [14] [15], the suggested approach produces a better avalanche effect. Fig. 4 shows the SAC performance analysis graph for the suggested system and the current approach.



Sl No	Plain Data	Encrypted Data	Balanced Output		Hamming Distance	Avalanche Effect %
			No. of 1's	No. of 0's		
1	Security	zùð(\q	35	29	21	54.6875
2	Networks	øJDÈ}ÀÑ	29	35	33	53.1250
3	RJ45JACK	qÕÚQCÅTÔ	31	33	32	53.1250
4	FUNCTION	ÕUª9ifÚ	36	28	27	56.2500
5	FRAGMENT	¾□°□h	30	34	38	54.6875
6	ZXQRASPK	¶iz\°/	33	31	36	60.9375
7	INVERTER	73's¾ý	37	27	35	57.8125
8	ELECTRON	ä□&J□,	28	36	35	57.8125

Table no. 2 : encrypted data, balanced output and hamming distance of the proposed method



Encryption Methods	Avalanche Effect
Piotr Mroczkowski et.al [12]	48
K. Kazlauskas et.al [13]	48.22
Fatma Ahmed et.al [14]	51.43
Mona Dara et.al [15]	53.9
Proposed Method	56.054

Table no. 3: avalanche effect comparison of proposed system with existing methods

The suggested system generates a decent SAC with enhanced performance.

V. CONCLUSION

A novel cryptosystem based on random numbers is created in this proposed study to enable safe communication in cellular networks. The KASUMI block cipher and chaos-based random numbers are used to create the desired model. Lorenz equations based on chaos are used to create random numbers. To make S-Boxes, random numbers are generated. These S-Boxes are employed in the process of encryption. For encryption, the KASUMI method is employed. The NIST test suite is used to verify the produced random numbers' unpredictability, and security parameters are used to assess the encrypted data's quality. With increased efficiency, the suggested method's outcomes meet the standard results rate. With minimal computing complexity, the suggested approach helps to improve secure communication in cellular networks. The suggested approach is contrasted with the avalanche effect approaches now in use. It demonstrates that the suggested approach outperforms current systems in terms of avalanche impact and efficiency.

REFERENCE

[1]. Ihsan Cicek, Ali Emre Pusane and Gunhan Dundar, "A Novel Design Method for Discrete Time Chaos Based True Random Number Generators", Elsevier, Vol. 47, No. 1, pp. 38-47, 2014.

[2]. Paul Tobin, Lee Tobin, Michael Mc Keever and J. Blackledge, "Chaos-Based Cryptography for Cloud Computing", In Signals and Systems Conference (ISSC).IEEE, pp. 1-6, 2016.

[3]. Wentan Yi and Shaozhen Chen, "Multidimensional Zero-Correlation Linear Cryptanalysis of the Block Cipher KASUMI", IET Information Security, Vol.10, No. 4, pp. 215-221, 2016.

[4]. Yazdeen Abdulmajeed Adil, Zeebaree Subhi RM, Sadeeq Mohammed Mohammed, Shakir Fattah Kak , Ahmed Omar M., and Zebari Rizgar R. "FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review." Qubahan Academic Journal 1, no. 2 (2021): 8–16.

[5]. Bisht Neeraj, Pandey Bishwajeet, and Budhani Sandeep Kumar. "Comparative performance analysis of AES encryption algorithm for various LVCMOS on different FPGAs." World Journal of Engineering (2022).

[6]. Alkamil Arkan, and Perera Darshika G. "Towards dynamic and partial reconfigurable hardware architectures for cryptographic algorithms on embedded devices." IEEE Access 8 (2020): 221720–221742.

[7]. Sala Della, Riccardo Davide Bellizia, and Scotti Giuseppe. "A novel ultra-compact fpga puf: The dd-puf." Cryptography 5, no. 3 (2021): 23.

[8]. Roy Debapriya Basu, Bhasin Shivam, Nikolić Ivica, and Mukhopadhyay Debdeep. "Combining puf with rluts: a two-party pay-per-device ip licensing scheme on fpgas." ACM Transactions on Embedded Computing Systems (TECS) 18, no. 2 (2019): 1–22.

[9]. Yasir Ning Wu, Zhang. "Compact hardware implementations of MISTY1 block cipher." Journal of Circuits, Systems and Computers, 2018, vol. 27, no. 3, pp. 14

[10]. Yasir, Wu Ning, Ali Zain Anwar, Shaikh Muhammad Mujtaba, Yahya Muhammad Rehan, and Aamir Muhammad. "Compact

and high-speed architectures of KASUMI block cipher." Wireless Personal Communications 106 (2019): 1787–1800.

[11]. Yamamoto Itoh, Yajima. “Compact architecture for ASIC and FPGA Implementation of KASUMI Block Cipher.” IEICE Transactions on fundamentals of Electronics, Communications and Computer Sciences. 2011, vol. E94-A, p. 2628–2638.

[12]. Gupta Chattopadhyay, Khalid . “Designing Integrated Accelerator for Stream Ciphers with Structural Similarities.” Cryptography and Communications-Discrete Structures Boolean Functions and Sequences, 2013, vol. 5, no. 1, p. 19–47.

[13]. Manz Olaf. "Symmetric Ciphers." In Encrypt, Sign, Attack: A compact introduction to cryptography, pp. 19–51. Berlin, Heidelberg: Springer Berlin Heidelberg, 2022.

[14]. Ntantogian Christoforos, Veroni Eleni, Karopoulos Georgios, and Xenakis Christos. "A survey of voice and communication protection solutions against wiretapping." Computers & Electrical Engineering 77 (2019): 163–178.

[15]. Stangherlin, Kleber, Zhuanhao Wu, Hiren Patel, and Manoj Sachdev. "Design exploration and security assessment of puf-on-puf implementations." arXiv preprint arXiv:2206.11840 (2022).