A New Key Generation Technique based on Neural Networks for Lightweight Block Ciphers

Sohel Rana[1*], M. Rubaiyat Hossain Mondal[2]
Institute of Information and Communication Technology
Bangladesh University of Engineering and Technology
Dhaka 1205, Bangladesh

A. H. M. Shahariar Parvez[3]
Department of Computer Science and Engineering
Dhaka University of Engineering and Technology
Dhaka 1700, Bangladesh

***Abstract*—In recent years, small computing devices used in wireless sensors, radio frequency identification (RFID) tags, Internet of Things (IoT) are increasing rapidly. However, the resources and capabilities of these devices are limited. Conventional encryption ciphers are computationally expensive and not suitable for lightweight devices. Hence, research in lightweight ciphers is important. In this paper, a new key scheduling technique based on neural network (NN) is introduced for lightweight block ciphers. The proposed NN approach is based on a multilayer feedforward neural network with a single hidden layer with the concept of nonlinear activation function to satisfy the Shannon confusion properties. It is shown here that NN consisting of 4 input, 4 hidden, and 4 output neurons is the best in key scheduling process. With this architecture, 5 unique keys are generated from 64 bit input data. Nonlinear bit shuffling is applied to create enough diffusion. The 4-4-4 NN approach generates the se-cure keys with an avalanche effect of more than 50 percent and consumes less power and memory, thus ensuring better performance than that of the existing algorithms. In our experiments, the memory usage and execution cycle of the NN key scheduling technique are evaluated on the fair evaluation of lightweight cryptographic systems (FELICS) tool that runs on the Linux operating system. The proposed NN approach is also implemented using MATLAB 2021a to test the key sensitivity by the histogram and correlation graphs of several encrypted and decrypted images. Results also show that compared to the existing algorithms, the proposed NN-cipher algorithm has lower number of execution cycles and hence less power consumption.**

***Keywords*—Lightweight cryptography; IoT; resource limited devices; neural network; avalanche effect; FELICS; MATLAB**

I. Introduction

Lightweight cryptography [1] is that the scaled-down version of traditional cryptography which target is to provide security for devices which resource capacity is restricted. While thinking in computer communication all people want to prevent his/her message from the malicious person as well as the valid one must receive/decrypt original message easily. However, there is a clear trade-off between lightweight and security of ciphers: Hence, a decent level of security is often achieved in such sort of resource-constrained devices. In recent years, the research community has been focusing on designing cryptographic primitives which are suited to those resource-constrained devices [2]. Conventional cryptographic algorithms like RSA [3] mostly perform well in powerful devices; therefore, lightweight algorithms do not seem to be necessary for them. Resources like read-only memory (ROM), random access memory (RAM), processing speed, and battery power are limited for resource-constrained devices like Embedded systems, radio frequency identification tags (RFID), and sensor networks. Hence, the lightweight block ciphers become essential to ensure the security of these devices. Different types of cryptographic algorithms like Advanced Encryption Standard (AES) [1], Data encryption standard (DES) [1], PRESENT [4], etc. are used for resource-constrained devices. In most resource-limited devices, lightweight block ciphers use such design architectures which will ensure enough security while keeping execution cycles as less as possible. Most of the ciphers follow the Feistel Architecture like Secure Internet of Things (SIT) [5], SIMON [6], Speck [6], etc. or by Substitution-Permutation Network like PRESENT [4], AES [7], etc. or by using both Architectures like DES, SIMON [6] to supply enough Shannon's confusion and diffusion properties in cipher text. Key scheduling in the block ciphers should perform in a secure way because the security of the ciphers depends on the secret keys which are used in every round of a block cipher. To make round keys strong, different complex number theories like modular arithmetic, prime factorization, Euclidian algorithms, etc. are applied in key generation techniques that end in hamper the performance of resource-limited devices. Good key scheduling must have two properties; randomness to generate unique keys and a high avalanche effect to ensure high key sensitivity. A single bit change in the key should change at least 50% bit in the cipher-text so that an attacker cannot easily predict a plain-text or keys through a statistical attack of encrypted message. That type of effect in cipher text is regarded as an avalanche effect. To implement a strong cipher, the avalanche effect should be considered as one of the primary design objectives.

A. Motivation

To research on cryptography is a challenging and interesting topic. Cryptography is the heart of secure data transmission. It includes complex mathematics, advanced programming, advanced number theories, etc. With the increasing usage of resource-constrained devices, lightweight block ciphers will be essential to provide security for those devices in near future. HP investigate that above seventy percent of re-source-limited devices are vulnerable to attacks [8]. There is a trade-off between the safety and performance of a low-powered small computing device. Since most of the cipher proposed to date is based on [1] complex number theory

*Corresponding Author

i.e., Modular arithmetic, Prime factorization, GCD testing algorithms, etc. As conventional ciphers use complex number theory to meet (Avalanche Effect) [1] Shannon confusion and diffusion properties so these ciphers generally become computationally expensive that hinders the performance of resource-limited devices. For that reason, it becomes challenging to implement these heavy algorithms in small computing devices for ensuring security. Hence, we focus on developing a simple and less power-consuming key generation technique using feedforward neural networks [9] (NN).

B. Contribution

This paper presents a NN-based key generation technique for lightweight block ciphers to provide the lightweight devices' security and meet the challenges of limited resource utilization. This paper puts NNs into the key scheduling process to generate strong round keys. This is done to achieve a more avalanche effect: more than 50% of the output bit is changed for a single bit change in input. This property is also called Shannon confusion. In the used NN, every bit in the input neuron is connected to all the output neuron bits. So, a change of a single bit can affect all bits in the output. It has addressed nonlinearity also since the sigmoid activation function is applied in hidden layer of proposed NN architecture. The proposed technique is computationally lightweight because NN uses simple mathematical operations like addition and multiplication to generate output. It also ensures nonlinearity as NN uses nonlinear activation functions like [10] sigmoid and Step Function. Normally this nonlinear transformation is done by S-BOX [11] like in AES [7], DES [7]. We faced few challenges to put the NN in the key generation technique. These are as follows.

- The choice of a NN to be used.
- The size of the input, hidden, and output neurons.

We evaluated 6 different feedforward NN architectures: NN4-4-4, NN4-3-4, NN4-2-4, NN4-1-4, NN4-5-4, and NN4-6-4. Among these, NN4-4-4 and NN4-3-4 perform better others. The performance of NN4-5-4 is also good; however it has higher computational complexity. We used 4-4-4 architecture as its average avalanche effect is higher than NN4-3-4, although the computational power of 4-4-4 is little more expensive than NN4-3-4. We used MATLAB 2021a to test the randomness by entropy, correlation, and histogram and key sensitivity i.e., Avalanche effect by encrypting images. Thus, the proposed algorithm ensures enough security and consumes less power.

II. Background and Literature Review

Lightweight devices have security vulnerabilities. Particularly, the devices used in IoT have security threats since these devices may remain without supervision for long hours [12]. Conventional ciphers require higher computations [13]. Most of the modern cryptographic algorithms proposed are based on complex number theory like [3] RSA, [1] ElGamal, and SPN [11] network like AES and [11] Fiestel architecture like [7] DES. The primary focus of using these primitives is to create keys and ciphertext more secure by ensuring more avalanche effect i.e., more confusion and diffusion in the ciphertext. The main problem with these primitives is computation-ally expensive, which hampers the performance of resource-limited devices if implemented.

A. NN in Cryptography

The NN is a core concept of computer science that can be implanted into cipher to achieve a higher avalanche effect in cryptography. As every neuron of input is connected to each neuron of output so a change of a single bit in input can affect all bits (neurons) of output. The NN also provides nonlinear transformation like confusion by using nonlinear activation functions like sigmoid. We can fix the output of a neuron by training the NN for different weights. Backpropagation is a good choice to train the NN. After that, a trained network with fixed weights can be implanted to cipher to make enough confusion in the ciphertext. Since NN is also a one-way transformation, reverse engineering attack is not effective.

B. Other Related Works

The authors of article [9], proposed a block cipher based on NN for cryptography. They designed an encryption algorithm based on NN to provide security in resource-constrained devices. Authors applied a [17] backpropagation network as supervised learning to train the fully connected feedforward NN. They claimed that the NN performs consistently and unconditionally throughout the encryption as well as decryption process. However, their algorithm was computationally expensive due to the extra burden of the training process of NN.

In [14], the authors evaluate the performance and security of modern [18] lightweight ciphers like TEA, HIGHT, KATAN, and KLEIN which are instigated especially in resource-constrained devices. To evaluate the performance metrics like memory and power consumption, the authors used the AtTiny45 microcontroller as a resource-limited device. Besides, they assessed the level of Shannon confusion and diffusion to testing the avalanche effect.

In [2], authors proposed a cipher for lightweight devices consisting of two core concepts of genetic algorithm, namely, two-point crossover and coin flip mutation. They also tested their proposed cipher on Fair Evaluation of Lightweight Cryptographic Systems (FELICS) to compare execution cycle and memory usages.

The author in [5] presented a symmetric cipher that combined together Feistel architecture and Substitution Permutation Network (SPN) to avail the linear and non-linear transformation in cipher text. They proposed a cipher that includes: key generation and the encryption process. The key scheduling section generates 5 unique keys by taking 64 bits master key as input from the user. After initial permutation, 64-bit input is grouped into 4 blocks each of which is 16-bit data in size. Every 4 blocks are fitted as input for f-function as shown in Fig. 1. A 4x4 metric is used to transform the output of the f-function. Here the only source of nonlinearity is the usage of the matrix. The F-function consists of two P-Boxes; used in key scheduling is just the linear transformation. We introduced NN-function in replace of F-function to provide both nonlinearity and high avalanche affect which results in high key sensitivity.

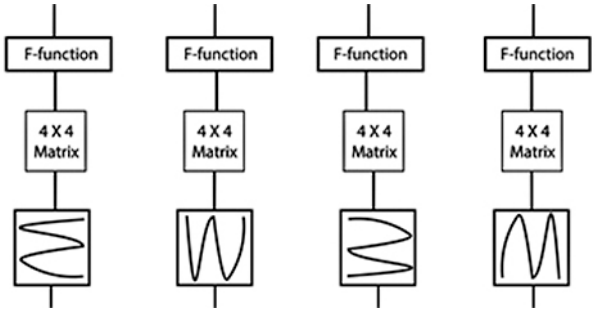

Fig. 1. 4x4 Matrix Format of Key Generation.

III. Proposed NN Approach

This section describes our proposed NN cipher approach. The proposed technique is a symmetric key block cipher based on a feedforward NN with a nonlinear activation function. The NN approach is suitable for low-powered and resource-limited devices. Using NN is to generate the keys strong enough by ensuring the avalanche effect more than 50%. We put the NN with nonlinear activation function into the key scheduling process to generate round keys used in different rounds of the encryption process. As every neuron of input is connected to each neuron of output so a change of a single bit in input can affect all bits (neurons) of output. These properties of NN meet the standard of avalanche effect that is more than 50%. We analyze the different sizes of NN to test the avalanche effect. Our experiment shows that NNs having too many or too few neurons in hidden layers than that of the input layer have less avalanche effect. We use NN consisting of 4 input neurons, 4 hidden neurons, and 4 output neurons in the key scheduling process. The size of the keys and plaintext is 64 bits.

A. Key Expansion

In a block cipher, the most fundamental component is the keys that are used to perform the encryption as well as decryption. If the key that was used to generate cipher text is compromised, the security is totally broken. Therefore, the illumination of the key should be as difficult as possible. To prevent data from different statistical attacks like chosen cipher text, chosen plain text, differential attack, etc. the sensitivity of the keys must be too high. Even if the attacker assumes the key that differs only a single bit from the original key, the result of decryption with that assumed key should be like cipher text. To ensure higher key sensitivity, the algorithm uses NN, with a nonlinear activation function for generating a key with an avalanche effect of more than 50%. Fig. 2 illustrates the process of the proposed key scheduling details.

The proposed algorithm needs 64-bit data as input to generate five unique keys for five rounds of the encryption process. These 64 bits of data are divided by 4 bits which generate 16 networks. Each network consists of 4-bit data. The proposed technique uses NN in 4 networks to create nonlinear transformation, and others are P and Q transformation to make enough diffusion in generated keys. A feedforward NN of NN 4-4-4 (4 input neurons, 4 output neurons, and 4 neurons of a hidden layer) is put on 1st, 6th, 11th, 15th networks, receipts 4 bits as input, and generates the 4-bit output with sigmoid and step activation function which is a nonlinear activation function. By using the nonlinear activation function in the hidden layer, the manufactured key is getting stronger with more confusion. Nonlinear bit shuffling is applied to create enough diffusion (linear transformation) in generated keys.

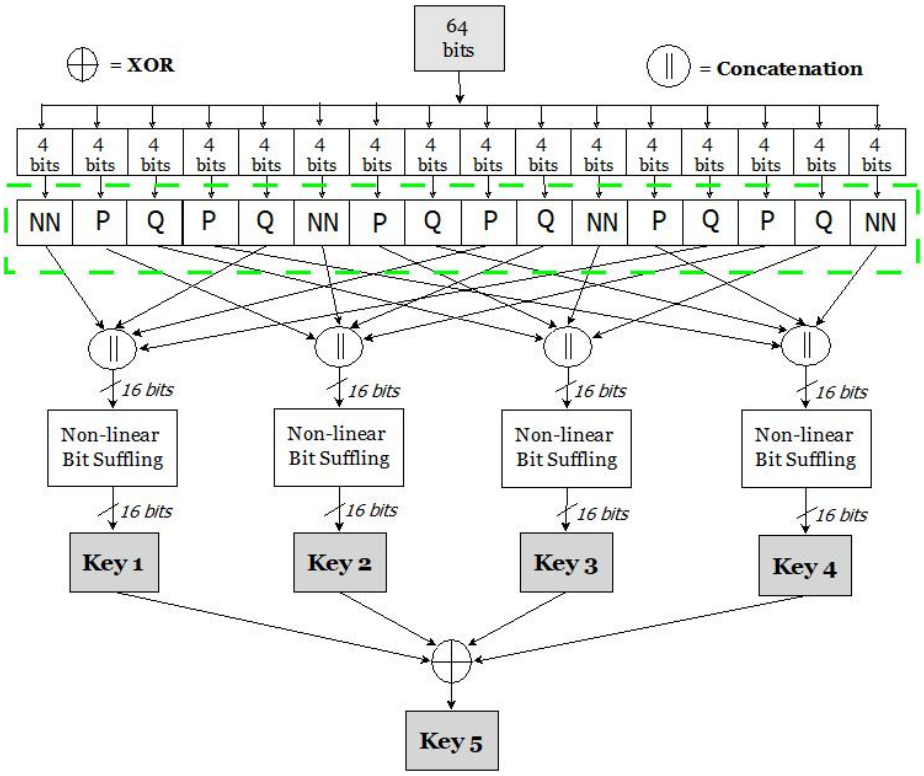

Fig. 2. NN based Key Scheduling Process.

After that the four 16-bit key (K1, K2, K3, K4) are generated, which consists of the output of four-bit networks. The next step is to apply XOR operation on every key K1, K2, K3, and K4 to generate the fifth key K5.

B. NN as a Function

In this paper, we use NN instead of the f-function [5], which is a combination of some linear transformation, i.e., permutation Box. The one-way properties of NN are more powerful to generate strong keys, and the avalanche effect of the proposed NN is more than 50 percent. This property of NN ensures the higher key sensitivity to protect different statistical cryptanalysis. Moreover, NN has less computational complexities because simple mathematical operations like addition and multiplication are used to produce output. Without NN, if anyone wants to generate the key with a higher avalanche effect, they must have to use complex number theory like prime factorization, Random numbers, and Modular arithmetic etc. For example, like RSA, large Prime numbers and modular exponential arithmetic are used to generate keys. Hence, there will be large computation power which is not acceptable for lightweight devices. In this proposed algorithm, a feedforward NN as shown in Fig. 3 is applied. The weights are trained with machine learning for a better output result. In a feedforward NN, input data travels in one direction only, passing through artificial neural nodes and exiting through output nodes. It has unidirectional forward propagation [9] but no backward propagation. Weights are static here. An activation function is fed by inputs which are multiplied by weights. The network has no cycles or loops.

Here, Xi =[X_1, X_2, X_3, X_4] are four neurons of input layer that join with their corresponding weights to 4 neurons of hidden layers Hj = [H_1, H_2, H_3, H_4] to generate 4 neurons of output layer O_k =[O_1, O_2, O_3, O_4]. The equation to calculate the each neuron of hidden layer is

$$h_j = \sum_{i=1}^{n} W_{ij} X_i + b_i \quad (1)$$

where h_j= hidden layer output, W_{ij}= weights, X_i= input bit and b_i=bias value.

The equation to calculate the each neuron of hidden layer is

$$O_k = \sum_{j=1}^{n} h_j W_{jk} + b_j \quad (2)$$

where O_k= desired output, h_j=hidden layer output, W_{ij}= weights and b_j= bias value of hidden layer.

C. Activation Function

Activation function produces nonlinearity into the output of a neuron. We used sigmoid function [10] is as activation function in the hidden layer.

$$\phi(H_j) = \frac{1}{1+e^{-H_j}} \; for \; j = 1, 2, 3, 4 \quad (3)$$

We used step function [10] as activation function in output layer because we need either 0 or 1 as output in output layer. The output is a certain value, A_1 for the case where the input summation has a values beyond a particular threshold, and A_0 when it is less than the threshold. The perceptron values were A_1 = 1 and A_0 = 0.

$$F(O_k) = \begin{cases} 1, & O_k > 0; \\ 0, & O_k \leq 0; \end{cases} for \; k = 1, 2, 3, 4 \quad (4)$$

From the truth table shown in Table I, it can be seen that the rate of bit changing is average 50% upper. This type of bit changing can make Shannon's confusion and diffusion easily. The security of the proposed key generation is standard for its image analysis because its correlation comes out in uniform patterns.

D. The Selection of Architecture NN 4-4-4

In cryptography, the keys must be generated by secure enough key scheduling algorithms. Generally, Different popular hashing algorithms like Secure Hash Algorithm (SHA), (Message Digest) MD5 etc., are used to generate rounds key to encrypt the plaintext. The most important issue in generated keys is the sensitivity of the keys. Any change in the key or the plaintext will lead to a significant change in the ciphertext so that an attacker cannot easily predict a plaintext or keys through a statistical analysis of ciphertext. This property is regarded as avalanche effect. To implement a strong cipher, avalanche effect should be considered as one of the primary design objective. The avalanche effect was identified by [1] "Shannon's property of confusion", however, the term was first mentioned by Horst Feistel. A strong encryption algorithm should always satisfy the [1] Avalanche effect > 50% criteria.

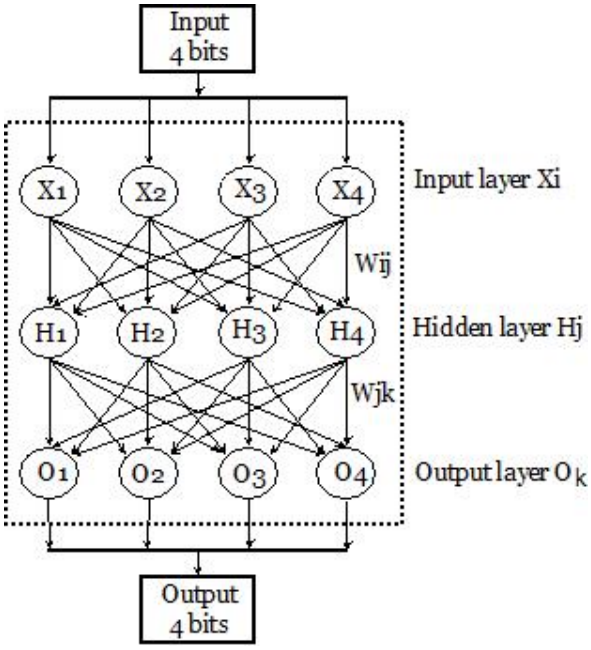

Fig. 3. Feed Forward NN (NN 4-4-4).

TABLE I. SAMPLE OUTPUT OF NN 4-4-4

Input	Output
1	1
1	0
1	0
0	1

An encryption algorithm that does not satisfies this property can favor an easy statistical attack. Considering the higher avalanche effect and lower computational power, we proposed the freed forward NN of 4 input neurons, 4 hidden neurons, and 4 output neurons for key scheduling process. As shown in Table II, we evaluate avalanche effect for different combinations of hidden neurons. We collect the outputs in python language. We do not consider all combinations of a single bit to test the avalanche effect but run the program for different combinations and put the average of collected outputs. According to the used weights, threshold values and activation functions, we got 53.125 % of avalanche effect (1-bit change) for architecture NN 4-4-4, NN 4-3-4, and NN 4-5-4 others are zeros. However, for 4-bit change, architecture NN 4-4-4 has the highest avalanche effect.

E. Computational Complexity of NN

In order to calculate the actual computational complexity of NN, it is necessary to know both the complexity for each operation and the number of operations. Assume that Computational Cost $C.C = M * 3 + A$ where M denotes the number of Multiplications and A denotes the number of addition in a NN. Here in the above equation, M is multiplied by 3 because in integer arithmetic multiplication is usually 3 times appreciably slower than addition. We do not consider the complexity of the activation function for simplicity of calculation.

TABLE II. EFFECT IN ROUND KEYS ON A SMALL CHANGE IN MASTER KEY

NN I/P - H - O/P	Affect in output for 1 bit changed in input (%)	Affect in output for 4 bit changed in input (%)	Computational cost
NN 4-4-4	53.125	59.375	32 Multiplications + 32 Additions
NN 4-3-4	53.125	56.25	24 Multiplications + 24 Additions
NN 4-2-4	00.00	00.00	16 Multiplications + 16 Additions
NN 4-1-4	00.00	00.00	8 Multiplications +8 Additions
NN 4-5-4	53.125	56.25	40 Multiplications + 40 Additions
NN 4-6-4	00.00	00.00	48 Multiplications +48 Additions

TABLE III. WEIGHED COMPUTATIONAL COST

NN I/P - Hidden - O/P	Computational Cost (Cycle)
NN 4-6-4	192
NN 4-5-4	160
NN 4-4-4	128
NN 4-3-4	96
NN 4-2-4	64
NN 4-1-4	32

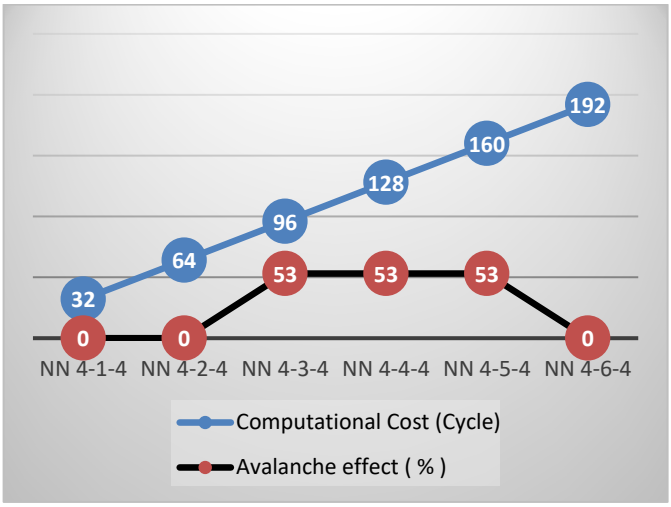

Fig. 4. Impact on Computational Power and Avalanche effect by each Increasing Hidden Layer.

Here M= 4 input weights * 4 inputs + 4 hidden weights * 4 output neurons = 32, similar for A = 32. So, C.C = 32 * 3 +32= 128. Accordingly, we calculated for all the NNs shown in Table III. The impact on computational power and avalanche effect by each increasing hidden layer are illustrated in Fig. 4. Horizontal axis represents the number of hidden layers for 4 input and output neurons.

The computational power cost increases linearly as we increase the hidden layer one by one. On the other hand, a number of architecture, namely NN 4-4-4, NN 4-3-4 and NN 4-5-4 have the avalanche effect of 53.125% (> 50%) which satisfy the standard of Shannon's property of confusion. We observed that as he hidden layers exceed the input or output neuron, avalanche effect remains the same and sometimes starts to fall. The performance of architecture with 5 hidden neurons is not improved. The optimal number of hidden [8] units should be smaller than the number of inputs. So, we selected architecture NN 4-4-4 to generate keys with high sensitivity.

F. Key Sensitivity for Ciphertext

To keep data secret from different statistical analyses like chosen ciphertext, chosen plain text, differential attack, encrypted ciphertext must change drastically (at least 50%) for the change of single bit of keys or plain text. We tabulated keys with single bit change and corresponding ciphertext for all mentioned architectures. Proposed architecture showed the avalanche effect more than 50%. All most entire avalanche effect depends on the NN in key scheduling. Table IV shows 6 pairs of cipher text for six different pair of main keys while a pair of keys differs only in single bit to each other.

G. Permutation (P and Q Table)

We used the P and Q tables as reported in the literature [5]. These tables perform linear transformations resulting in diffusion. However, we used P and Q table as per our new design of the key scheduling. The transformations made by P are shown in Table V.

TABLE IV. AVALANCHE EFFECT IN CIPHER TEXT FOR SINGLE BIT CHANGED IN KEY

NN I/P - H - O/P	Input Keys (64 bits) in hex	Cipher text(64 bits) in hex	Avalanche Effect (%)
NN 4-4-4	12 34 56 78 9a bc de fa	04 36 12 d2 97 8f 1b af	53.125
	12 34 56 78 9a bc de fb	48 d b8 55 2d 30 5b 80	
NN 4-3-4	12 34 56 78 9a bc de fa	04 36 12 d2 97 8f 1b af	53.125
	12 34 56 78 9a bc de fb	48 d b8 55 2d 30 5b 80	
NN 4-2-4	12 34 56 78 9a bc de fa	d3 29 51 da ec 8f c8 b4	00.00
	12 34 56 78 9a bc de fb	d3 29 51 da ec 8f c8 b4	
NN 4-1-4	12 34 56 78 9a bc de fa	d3 29 51 da ec 8f c8 b4	00.00
	12 34 56 78 9a bc de fb	d3 29 51 da ec 8f c8 b4	
NN 4-5-4	12 34 56 78 9a bc de fa	04 36 12 d2 97 8f 1b af	53.125
	12 34 56 78 9a bc de fb	48 d b8 55 2d 30 5b 80	
NN 4-6-4	12 34 56 78 9a bc de fa	d3 29 51 da ec 8f c8 b4	00.00
	12 34 56 78 9a bc de fb	d3 29 51 da ec 8f c8 b4	

Here, Plain text used is 0xab cd 12 34 87 65 01 35

TABLE V. P TABLE

Input key(k_i)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Generated key(k_i)	3	F	E	0	5	4	B	C	D	A	9	6	7	8	2	1

The transformations made by Q are shown in the Table VI.

TABLE VI. Q TABLE

Input key(k_i)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Generated key(k_i)	9	E	5	6	A	2	3	C	F	0	4	D	7	B	1	8

H. Non-Linear Bit Shuffling

We used the nonlinear bit-shuffling as reported in the literature [2]. In the nonlinear bit-shuffling, a concatenated 16-bit is transferred to each block of non-linear bit shuffling. After that, a random number is computed from the input of 16-bit data which is again logically combined with 16-bit input by performing a bitwise XOR operation. The output of the XOR operation is transferred to bit shuffling as well as to the perfect shuffling block sequentially to create enough diffusion in generated keys.

I. Encryption and Decryption Process

We used the encryption algorithm which was proposed by [2]. The encryption process consists of Feistel architecture with G-function [2] which is based on the concept of two operators of genetic algorithms: mutation and crossover. Fig. 5 illustrates the flow of operation for a single among five rounds. The encryption process takes a block of 64-bit plain messages as input at a time. The first round uses the first key K1 generated by NN based key generation technique and then K2, K3, K4, K5 for 2nd, 3rd,4th, and 5th rounds sequentially.

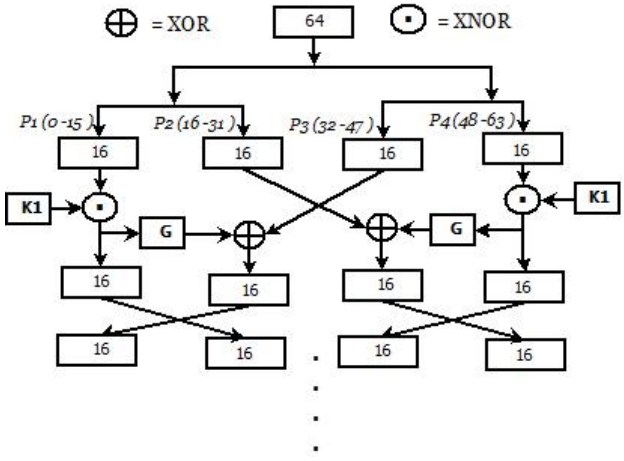

Fig. 5. A Single Round of Encryption Process.

$$Ro_{i,j} = \begin{cases} Px_{i,j} \odot K_i \; ; \; j = 1 \; and \; 4 \\ Px_{i,j+1} \oplus EG_{li} \; ; \; j = 2 \\ Px_{i,j-1} \oplus EG_{ri} \; ; \; j = 3 \end{cases} \quad (5)$$

The 64-bit message is equally split into four 16 bit segments. According to the Feistel structure, swapping, XOR, XNOR operations are performed among the split blocks to increase the avalanche effect in cipher text. An XNOR operation is performed between round key and left as well as rightmost blocks separately. Then the output of the XNOR operation is feed to G-function as input. The 4th block and output of the left G-function are again XORed and the 2nd block and output of the right G-function are XORed as well. After that, a swapping operation is executed among the 4 blocks except for the last round. The equation (5) represents the process of how a single rounds encrypt plaintext into cipher text. Finally, every four blocks are combined together to generate a block of 64-bit cipher text. The decryption process is the opposite of the encryption process. This time last key is used first.

J. G-Function

We replicated the G-function from the earlier research work. The G-function [2] is based on the concept of two operators of genetic algorithms: mutation and crossover. This function takes 16 bits as input and first, split equally into two eight-bit segments. Both two 8 bit data are transformed by using a substitution box that is called S-Box. After performing a two-point crossover to both outputs of S-Box, a coin flip mutation operation occurs. Finally, the 16-bit output is generated.

IV. Experimental Setup

The proposed approach is initially written in a popular structure language C. We used CodeBlocks as IDE. The coded cipher is not dependent on any machine to be executed. We also used a benchmark tool 'Fair Evaluation of Lightweight Cryptography Systems (FELICS) to measure memory usage and execution cycles that run on Linux Ubuntu. The FELICS tool is open access and free to install. We also implemented our proposed NN approach in MATLAB 2021a to evaluate the security strength of keys. The proposed NN approach is evaluated based on key sensitivity, execution cycles, bridge histogram, and correlation histograms. We also assessed the NN cipher in terms of memory usage and clock cycles to generate keys, cipher text, and regenerate plaintext.

A. Evaluation Parameters

The security strength of the proposed algorithm is tested based on key sensitivity, execution cycles, bridge histogram and correlation histograms. We also assessed the memory utilization and execution cycles for key generation, encryption and decryption of this algorithm. The FELICS [15] provides a command-line interface like GCC (GNU Compiler Collection) to test and build any lightweight cryptographic code. They provide documentation to facilitate the implementation as shown in Fig. 6. We can compile our implementation and test whether ours is runnable in FELICS or not. It provides three scenarios against which we can test our code.

We used FELICS to collect clock cycles required for the key generation, encryption, and decryption process of different reported ciphers along with the proposed cipher. We measure the program memory, RAM, and actual code size as well. Table VII shows the comparative results of different ciphers for AVR architecture. It can be seen that among the methods considered, the proposed NN method requires the lowest total execution cycles. Though other ciphers like HIGHT [16] have fewer cycles for key generation, the overall execution cycle of NN is fewer than that of others. The NN cipher needs the lowest RAM to execute and hence, it is memory efficient also.

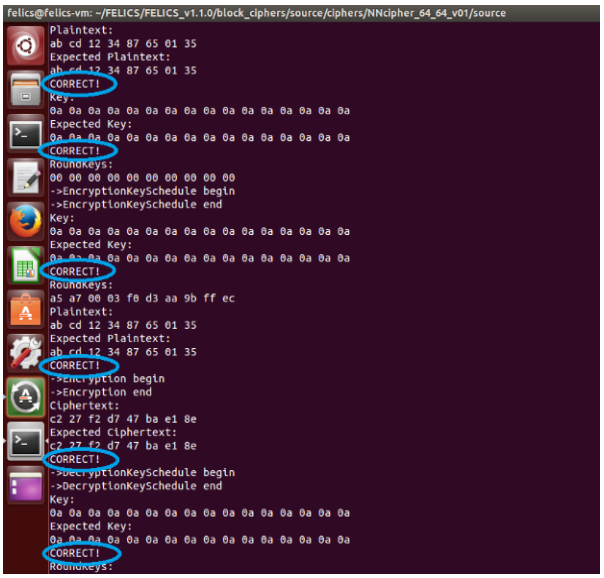

Fig. 6. Testing the Proposed Approach in FELICS.

TABLE VII. Execution Cycles of Ciphers on AVR Architecture

CIPHER	Device	Block size	Key size	CODE SIZE	RAM	Cycles (Key generation)	Cycles (encryption)	Cycles (decryption)
AES[7]	AVR	128	128	23090	720	3274	5423	5388
HIGHT[16]	AVR	64	128	13476	288	1412	3376	3401
LEA[14]	AVR	128	128	3700	432	4290	3723	3784
PRESENT[4]	AVR	64	80	1738	274	2570	7447	7422
RC5	AVR	64	128	20044	360	26793	4616	4652
Simon[6]	AVR	64	96	1370	188	2991	1980	1925
Speck[6]	AVR	64	96	2552	124	1509	1179	1411
SIT[5]	AVR	64	64	826	22	2130	876	851
G-cipher[2]	AVR	64	64	1228	34	1630	792	789
Proposed NN-Cipher	AVR	64	64	1228	34	1483	792	789

Fig. 7 presents bar chart comparisons among various reported ciphers with NN approach. For each ciphers, bar chart shows the status of required clock cycles to generate keys, cipher text from plaintext, plaintext from cipher text, as well as overall execution cycles. The chart shown in Fig. 7 clearly demonstrates that the NN cipher executes in less clock cycles, improving over the other reported ciphers especially than SIT [5] and G-cipher [2] which we followed most. So, the proposed NN cipher is more power efficient than that of other ciphers.

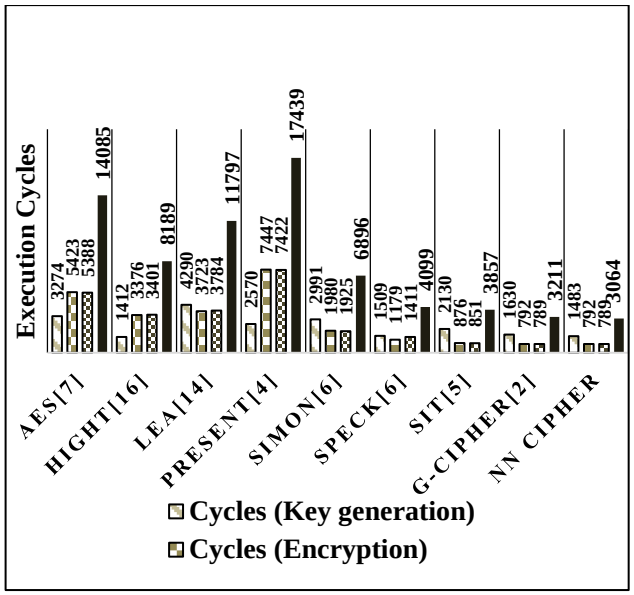

Fig. 7. Execution Cycle of Ciphers for Hardware Implementation.

B. Analysis of Key Sensitivity

The NN approach is also demonstrated in MATLAB® which encrypts an image and then decrypts the image with the correct key for a visual observation key sensitivity. After that the images are decrypted by using a wrong key with a single bit alteration from the original key. This is also a test for the avalanche effect of the keys. This is also a test for the avalanche effect of the keys. The decryption is non-recognizable if even one bit changes in the original keys. Fig. 8 shows that for NN-cipher, the encrypted images can only be decrypted with the correct key.

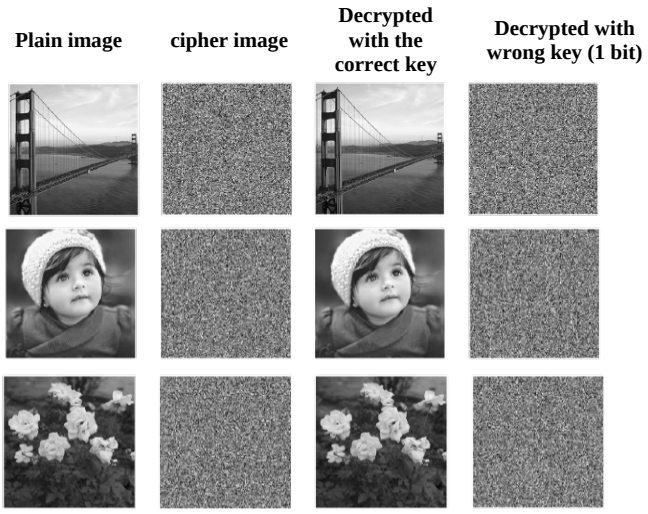

Fig. 8. Analysis of Key Sensitivity.

C. Histogram and Correlation Comparisons

Fig. 9 presents the bridge histograms of original and encrypted images. The vertical line indicates the pixel available in the image, and the horizontal axis refers to image intensity. The histogram of the image shows the uniform relationship that ensures the strength of the cipher image. So any statistical attacks will not be effective to predict the plain image from the cipher image without using the correct keys.

Fig. 10 shows the correlation graph of the considered original images and the encrypted images. The correlation graph of plain image shows linear relationship that is higher positive correlated value. However, the correlation graph of cipher image shows highly randomness that is the negative values. Hence, the negative correlation values of encrypted images indicate the strong security strength of proposed cipher.

D. Power Consumption

For calculating the total power consumed by an algorithm on a particular device, first we need the execution cycle of that algorithm. By using the following equation (6), we can compute the power consumption of an algorithm on a particular device:

$$E = I * V_{cc} * N * \tau \quad (6)$$

Where, V_{cc} denotes the operating voltage and I indicates operating current used up for T seconds (unit in Ampere). τ refers to the clock period as well as N indicates the required number of execution cycle. If f be the operating frequency of the particular device in Hertz then we can calculate the time period of the particular device that is $\tau = {}^{1}/_{f}\ sec/cycle$.

According to the absolute maximum rating (AMR) of dataset, usually maximum operating voltage of [19] Atmel Atmega128 is 5V, Maximum current is 40mA and operates at 16 MHz. Fig. 11 demonstrates the of energy consumption of different reported ciphers along with the proposed NN approach for a single block of data. The bat chart shows that NN approach consume less power than that of others.

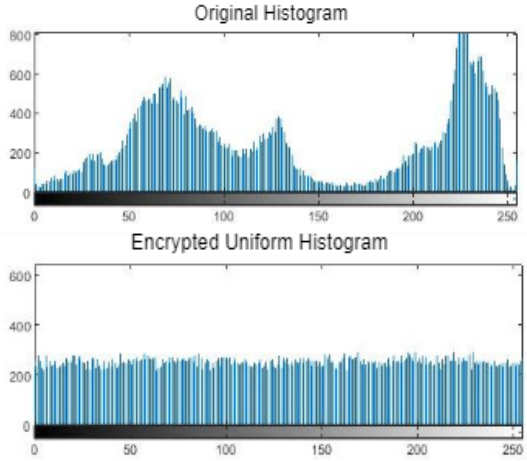

Fig. 9. Bridge Histogram.

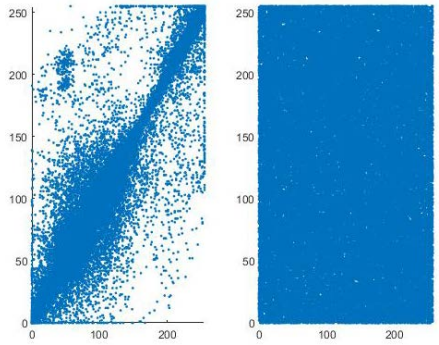

Fig. 10. Bridge Correlations for Encrypted and Decrypted Image.

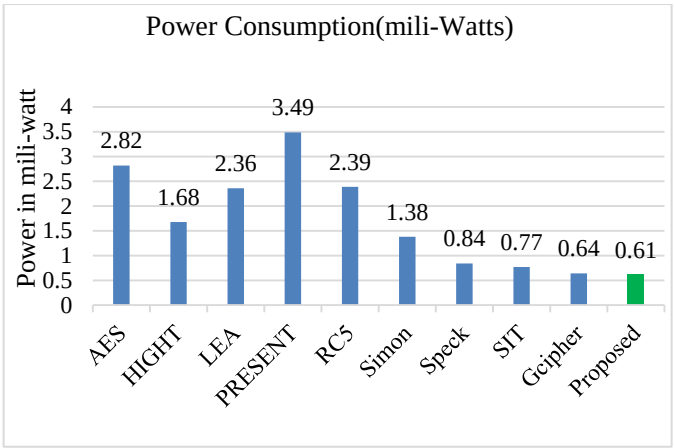

Fig. 11. Energy Consumption Comparison of Ciphers.

V. Conclusion and Future Work

The security, as well as performance of resource-limited devices, is an important issue. For this purpose, a lightweight cryptographic algorithm using NN is proposed in this paper. This NN-cipher algorithm has lower key generation cycles and less power consumption than the existing ciphers. The bridge histogram and the bridge correlation plot indicate that the NN-cipher can reliably encrypt images. Moreover, the key sensitivity results indicate that for NN-ciphers, encrypted images can only be successfully decrypted using the actual key. Hence, the proposed cipher will be an excellent solution of security for those devices that are resource-limited. Besides, We intend to perform more mathematical analysis as well as hardware implementation on our proposed algorithm to investigate its security strength as future work. More evaluation metrics like the randomness of generated keys using the Chi-square test can be considered for further evaluation.

References

[1] William Stallings "Cryptography and Network Security Principles and Practices", Fourth Edition, Publisher: Prentice Hall, November 16, 2005.

[2] Sohel Rana, Saddam Hossain, Hasan Imam Shoun and Dr. Mohammod Abul Kashem, "An Effective Lightweight Cryptographic Algorithm to Secure Resource-Constrained Devices" International Journal of Advanced Computer Science and Applications (IJACSA), 9(11), 2018. DOI: 10.14569/IJACSA.2018.091137.

[3] Ekhlas Abbas Albahrani,Tayseer Karam Alshekly . "Image Cipher System Based On RSA And Chaotic Maps,Hindawi, Security and Communication Networks." Security and Communication Networks, Article ID:5586959, pp. 18 Pages, https://doi.org/10.1155/2021/5586959 .(2021).

[4] Kostas Papapagiannopoulos."High throughput in slices: the case of PRESENT, PRINCE and KATAN64 ciphers". Radboud University Nijmegen, Department of Digital Securi-ty.2016.

[5] Muhammad Usman, Irfan Ahmed, M. Imran Aslam, Shujaat Khanand Usman Ali Shah. "SIT: A Lightweight Encryption Algorithm for Secure Internet of Things" Iqra Universi-ty, Defence View and Department of Electronic Engineering (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 8, No. 1, 2017.

[6] Tomer Ashur, Atul Luykx. "An Account of the ISO/IEC Standardization of the Simon and Speck Block Cipher Families." Security of Ubiquitous Computing Systems.Edited by © 2020 Springer Nature Switzerland AG. Part of Springer Nature, Vol License CC BY 4.0, pp. 63-78 Switzerland, DOI: 10.1007/978-3-030-10591-4_4 2021.

[7] Biao Xing1, DanDan Wang1, Yongquan Yang1, Zhiqiang Wei2, Jiajing Wu1, Cuihua He . "Accelerating DES and AES Algorithms for a Heterogeneous Many-core Processor." International Journal of Parallel Programming (2021) , Int., pp. 49:463–486 , https://doi.org/10.1007/s10766-021-00692-4. (2021).

[8] S. A. Kumar, T. Vealey, and H. Srivastava, "Security in Internet of Things: Challenges, solutions and future directions", in 2016 49th Hawaii International Conference on Sys-tem Sciences (HICSS), IEEE, 2016, pp.5772-5781.

[9] Eva Volna , Martin Kotyrba,Vaclav Kocian, Michal Janosek, CRYPTOGRAPHY BASED ON NEURAL NETWORK, Proceedings 26th European Conference on Model-ling and Simulation ©ECMS Klaus G. Troitzsch, Michael Möhring.

[10] Chigozie Enyinna Nwankpa, Winifred Ijomah, Anthony Gachagan, And Stephen Mar-shall, "Performance Analysis Of Various Activation Functions In Artificial Neural Net-works",Journal Of Physics Conference Series 1237:022030, June 2019; DOI: 10.1088/1742-6596/1237/2/022030.

[11] Sohel Rana, Wadud, Ali Azgar, Dr. M Abul Kashem, "A Survey Paper of Lightweight Block Ciphers Based on Their Different Design Architectures and Performance Metrics" International Journal of Computer Engineering and Information Technology, June 2019, Volume 11, Issue 6.

[12] P. Wang, Professor S. Chaudhry, S. Li, T. Tryfonas and H. Li, "The internet of things: a security point of view", Internet Research, vol. 26, no. 2, pp. 337-359, 2016.

[13] S. Wang, Z. Zhang, Z. Ye, X. Wang, X. Lin, and S. Chen, "Application of environmen-tal internet of things on water quality management of urban scenic river", International Journal of Sustainable Development & World Ecology, vol. 20, no3, pp. 216-222, 2013.

[14] Vikash Kumar Jha," Cryptanalysis of Lightweight Block Ciphers" Aalto University School of Science Degree Programme of Computer Science and Engineering, Master's Thesis, November 18, 2011.

[15] D. Dinu, A. Biryukov, J. Großschädl, D. Khovratovich, Y. L. Corre, L. Perrin, "FELICS – Fair Evaluation of Lightweight Cryptographic Systems", University of Luxembourg, July 2015.

[16] Bohun Kim,Junghoon Cho,Byungjun Choi,Jongsun Park,and Hwajeong SeoHindawi."Compact Implementations of HIGHT Block Cipher on IoT Platforms. Hindawi, Security and Communication Networks, Volume 2019, Article ID 5323578.

[17] Tope Komal, Rane Ashutosh, Rahate Roshan, S.M.Nalawade, Encryption and Decryp-tion using Artificial Neural Network, International Advanced Research Journal in Sci-ence, Engineering and Technology, Vol. 2, Issue 4, April 2015,

[18] Kerry A. McKay, Larry Bassham, Meltem Sönmez Turan, Nicky Mouha. Report on Lightweight Cryptography, National Institute of Standards and Technology,USA, March 2017.

[19] Utsav Banerjee, Lisa Ho, and Skanda Koppula."Power-Based Side-Channel Attack for AES Key Extraction on the ATMega328 Microcontroller". Massachusetts Institute of Technology. ResearchGate. December 201.