



IEVCA: An efficient image encryption technique for IoT applications using 2-D Von-Neumann cellular automata

Satyabrata Roy¹ · Manu Shrivastava¹  · Chirag Vinodkumar Pandey¹ · Sanjeet Kumar Nayak² · Umashankar Rawat¹

Received: 27 April 2020 / Revised: 10 September 2020 / Accepted: 15 September 2020 /
Published online: 1 October 2020

© Springer Science+Business Media, LLC, part of Springer Nature 2020, corrected publication 2021

Abstract

Present era is marked by exponential growth in transfer of multimedia data through internet. Most of the Internet-of-Things(IoT) applications send images to cloud storages through internet. However, in sensitive applications such as healthcare, defense, etc., these images should be encrypted before transmission through insecure public channels to gateway fog nodes. Conventional encryption algorithms cannot be used there due to the resource constraint characters of IoT devices. Here, Cellular Automata (CA) based encryption algorithms can be used because of their inherent simplicity in implementation in hardware, without affecting the capability of generating highly random sequences. In this paper, a lightweight, robust and secure image encryption technique has been proposed using 2-D Von-Neumann Cellular Automata (VCA), called IEVCA, which is lossless, correlation immune and has all the essential properties of a good image cipher. Additionally, the proposed technique passes all the randomness tests of DIEHARD and NIST statistical test suites. Moreover, several security and performance analyses of the IEVCA proved its efficiency and resistance against security attacks. Experimental results of the IEVCA show its better performance when compared to the existing encryption techniques.

Keywords Lightweight · IoT · Image encryption · 2-D Cellular automata · Block cipher

1 Introduction

The rapid outbreak of Internet of Things (IoT) is one of the astounding technologies that has made a high impact in the modern life and livelihood of people. IoT was invented by Kevin Ashton from Massachusetts Institute of Technology (1998) [20]. It is not a standalone technology, rather, it is a mixture of many other technologies, platforms, cyber-physical systems and so on and so forth. More specifically, it is an extended network of physical

✉ Manu Shrivastava
manu.shrivastava@jaipur.manipal.edu

objects or “things” combined with different components such as sensors, actuators, micro-controllers, many cyberunits, internet and other connectivity enablers so that “things” can transmit data among each other from anywhere and everywhere in the form of image, audio, video, signal, text etc.

IoT comes with broader challenges of synchronization among different platforms that are limited by their own constraints in terms of framework, connectivity, power consumption, bandwidth, communication range and compatibility. However, once developed, it can perform the herculean tasks of surveillance, healthcare monitoring, industry automation, agricultural automation, environment monitoring, etc., with minimum human intervention. The number of connected “things” are increasing expeditiously day by day and is expected to reach 29 billion by the end of 2020 [46].

Industry 4.0, also known as the fourth revolution of industry is also empowered by IoT applications [16]. It has made way for systematic deployment of advanced Information and Communication Technologies (ICT), Future oriented Techniques (FoT) and Intelligent Information Processing (IIP) for continuous monitoring and enhancement of various automatic processes. One of the major challenges in context of Industry 4.0 is providing security to the underlying independent applications. Industry 4.0 has wide range of contributions in industrial wireless sensor networks, smart healthcare, dairy industries, agricultural industries, automobile industries, defense research etc. [14, 15].

In recent times, with the advent of multimedia technologies, where images are playing important role in conveying large number of information very easily at a time, it is important to preserve the privacy of these data. Needless to say, these multimedia data are often sent through insecure public platform by camera sensors. It gives rise to the potential risk of the vulnerable data sent by these sensors [42].

In cloud enabled IoT infrastructure, at the perception layer or at the physical layer, tiny sensory devices gathers data in different forms and transfer them to the cloud storage via the Internet. These information is further processed at network layer and are used for analysis at application layer [45, 67]. However, interconnection of many cyber-physical units or “things” gives rise to various threats and attacks [19, 28]. The vulnerabilities of such interconnections where sensitive data exchanges happen at a rapid rate [71], get exposed to the adversaries. Apart from that, the sensors have their own constraints in terms of less memory size, low computing capability, short battery life, bandwidth, etc. [69]. As a consequence, conventional and widely used symmetric key block ciphers such as Advanced Encryption Standard (AES), Data Encryption Standard (DES) [51] or public key ciphers such as Rivest-Shamir-Adleman (RSA) [50], Diffie-Hellman [9] cryptosystems cannot be implemented because of their complex structures and heavy resource consumption. Hence, developing lightweight algorithms for providing security to these applications is of utmost importance.

An attacker can manipulate sensitive image data during its transmission from perception layer to the network layer, where these data are processed intermediately before sending them to cloud servers [32]. As mentioned earlier, the widely used ciphers such as AES, DES, RSA, Diffie-Hellman etc., can be implemented at the network layer and cloud empowered application layer. On the other hand, since perception layer consists of resource constrained sensory devices, there are not much available lightweight cipher for image encryption. Image encryption often requires chaotic map, DNA computing, quantum computing etc., for producing high degree of randomness in the cipher images at the cost of memory, computing power, battery life, etc [12, 25, 70].

There are image encryption techniques that use chaotic maps along with one-time keys [33, 34], chaotic maps along with DNA rules [35] or chaotic maps along with neural network [64] for efficiently generating cipher image from a given plain image. The cipher images contain good cryptanalysis attack resistance properties and they are robust against certain noise levels. All of these cipher images show good NPCR and UACI values when differential analysis is performed on them. But, majority of the techniques are resource consuming in nature. Consequently, although those techniques can be implemented for conventional applications that color image encryption, but these cannot be applied in IoT applications where resource constraints exist in the sensory devices.

In this work, an image encryption scheme called IEVCA has been developed and implemented in IoT applications that use camera sensors for surveillance purpose. The proposed cipher makes use of 2-D Von-Neumann Cellular Automata (2VCA) with five neighbors [8]. The 2VCA cells can store one bit value at a time. All the five neighbors of a cell can store one bit at a time. These neighbors together participate in changing the bit value from '1' to '0' or vice-versa at next time stamp [6]. When it is applied to an image, it shows complex and random arrangements of the image pixels. Moreover, it can exhibit the globally complex nature by using only at most five bits at a time. This characteristic of 2VCA makes it suitable for utilizing in developing lightweight image cipher for IoT applications. 2VCA cells can generate highly chaotic sequence that can be used in image encryption or pseudo random number generation [30]. Since, CA are highly convenient to be applied in hardware and consists of simple operations locally, the proposed scheme is lightweight and it can be easily implemented in sensor devices at physical level or perception layer.

1.1 Contribution

We have performed many standard analyses and randomness tests to check the performance of the proposed scheme against other state-of-the-art image ciphers. The comparison results have confirmed its capability to generate high degree of randomness in cipher image. Further, it is also proved that the proposed scheme can prevent many cryptanalysis attacks like brute-force attacks, known plaintext, chosen plaintext, known ciphertext and chosen ciphertext attacks. It makes IEVCA capable of being deployed for securing device to network communication. In short, the major contributions of this work are:

- IEVCA is a symmetric key block encryption technique that uses 2VCA rule vectors for encryption of color images captured by the camera sensors at the physical/perception layer. The encryption happens at the physical layer and decryption at the network layer. The encryption is done through efficient selection of rule vectors and iteration number.
- It is totally a lossless encryption scheme that uses pixel substitution of color images to achieve high amount of confusion and diffusion property. IEVCA has undergone majority of the analysis that are conventionally applied for any image cipher. The analyses include key space analysis, correlation coefficient, histogram, key sensitivity, differential analysis, image quality analysis etc., that show that the generated cipher image has low correlation, it achieves high amount of confusion and diffusion, the technique is highly key sensitive, as is expected from any conventional image cipher. In addition to these, IEVCA has undergone the NIST and DIEHARD randomness tests. The results are compared to other similar types of existing works. The comparison shows that IEVCA achieved expected level of p-values ensuring the presence of high randomness in cipher images as compared to the existing techniques.

- IEVCA is implemented using Raspberry Pi 3 and camera sensors. The execution time of the proposed scheme is also compared with state-of-the-art ciphers. The comparison result justifies that the performance of our scheme is better than in terms of execution time without compromising with the other essential security criteria. It makes IEVCA suitable for implementation in real time sensitive IoT applications.

1.2 Paper organization

The subsequent sections of this paper is organized as follows. Section 2 presents state-of-the-art of various image encryption techniques using 2VCA. Section 3 presents a brief description of the essential concepts required for understanding of the current work. Section 4 presents the proposed model and algorithms of this work. Section 5 presents the results of the proposed scheme. Section 6 presents various security and performance analyses of our scheme and the comparison with other existing works. Finally, Section 7 concludes the work by suggesting some future research directions.

2 Related works

The study of Cellular Automata started long ago by John Von-Neumann et al., who popularized it as an abstract model for studying self reproduction [56] along with his mathematician friend Stanislaw M. Ulam [4]. Since then, many researchers continuously put efforts to simulate complex behavior of biological systems through CA.

Later, 2-D CA characterization, reversibility and VLSI architecture for CA Machines (CAM) were proposed by [6, 26]. Dihidar et al. [10] proposed some exceptional rules of 2-D CA with matrix algebra. Afterwards, study of 2-D CA was extended to find issues in drawing state transitions, finding decidable properties, linear rules, analysis of 2-D CA rules in ternary fields [52] and many mathematical theorems and corollaries were derived. All of these works have ensured the capability of 2-D CA in generating complex patterns through simple local transition rules.

In [2], an image encryption scheme with authentication capability has been proposed by applying three different techniques – chaotic map, permutation-diffusion architecture and memory CA. Produced cipher images contain both confusion and diffusion properties. Additionally, their scheme can detect any tampering during transmission. In [61], the authors proposed an image cipher with intertwining chaotic map for achieving confusion and periodic-boundary reversible CA for achieving diffusion property. The authors have considered only highest 4 bits of the image pixels for diffusion as they contain maximum amount of information. The runtime and operational complexities of these methods are comprehensibly resource consuming. Hence, they cannot be implemented in sensory devices.

Chen et al. [7] proposed a 2-D CA architecture for VLSI implementation in hardware. The proposed cipher was constructed using re-configurable Von-Neumann 2-D CA. In these works, security analysis using different standard metrics, randomness tests in generated cipher images were not performed. In [48], authors have proposed 1-D CA based encryption methods both for images and ECG hash codes. In spite of having large key space that helps preventing naive brute-force attacks, their performances against different types of statistical analyses are not good, making them not suitable for implementation in sensitive IoT devices.

Application of 2-D CA for text compression was proposed by Khan et al. [27]. They designed an abstract algebraic model through matrices for periodic-boundary 2-D CA.

Further, a VLSI architecture for text compression was also proposed by the authors. An architecture of lightweight image cipher was proposed by Torres et al. [54]. This architecture was useful for implementing image encryption method in mobile devices with resource limitations. The proposed encryption method is a stream cipher based image cipher implemented using field programmable gate array (FPGA) devices.

Some novel image encryption techniques using DNA sequence, hybrid hyper chaotic systems, chaotic skew tent map and genetic operations are proposed in [1, 18, 39, 43, 60, 65]. Each technique have utilized CA for incorporation of permutation and diffusion. Their schemes are robust and capable of generating high quality cipher image when ample amount of computing resources are available. In [23], an image cipher is proposed using elementary CA rules. Different statistical tests and standard analyses proved their technique a good choice for image encryption.

In [44], the authors have presented a lightweight cipher that can be used for image encryption in resource constrained environment. It has significantly less execution time and can mitigate majority of security attacks such as known plaintext, chosen plaintext attacks. This scheme can be deployed with very low memory and with combination of static and dynamic keys.

In [29], a lightweight selective encryption scheme is proposed to encrypt the edge maps of medical images. This scheme uses chaotic map to generate a big key space. A one-time pad is also proposed for enciphering the detected image blocks. This scheme is lightweight and can prevent many security attacks, making it a good candidate to be implemented in IoT applications.

In [24], an unification technique is proposed for image compression and encryption in a single module. The unification is done through hyper-chaotic sequences and sparse representation of frames of any input image. This technique is able to provide a higher compression ratio (CR) compared to the state-of-the-art techniques. It also increases security level by using key-sequence in various steps of encryption. The output after unification is stronger than simple encryption techniques that use chaotic systems.

In [63], a visually secure encryption technique is proposed. This technique uses parallel compressive sensing (PCS) counter mode and embedding technique. In this technique, Logistic-Tent system is used to construct the measurement metrics. 3-D Cat maps are used to jumble the the order of embedded information. The scheme has achieved superior imperceptibility.

In [38] an image encryption scheme has been proposed using reversible CA. The scheme is a parallel block encryption scheme generating highly random cipher images. In [13] an image encryption scheme using a 2-D Tinkerball chaotic map, DNA sequences and CA has been proposed. Their proposed scheme is a gray image encryption scheme. In [68], a gray image encryption technique using quantum CA has been proposed. The invertible quantum CA operations enabled the scheme to achieve runtime complexity of $\theta(n)$ which is better than other schemes using quantum CA. The security analyses of all the schemes are good enough to be implemented in high-end computers, but not in resource limited sensory devices.

In [57] a fast image encryption algorithm based on parallel computing has been proposed. Here, cycle shift and permutation methods are combined to get good permutation results. Diffusion method is implemented using parallel computing and it shows good performance in terms of time and space complexities. Recently some image encryption algorithms have been proposed [58, 59] that use matrix semi-tensor product with a compound secret key. This key is produced by a Boolean network. Here, chaotic system has been used to generate

random key streams. The produced cipher images exhibit good security characteristics when compared to other existing image encryption techniques.

In the state-of-the-art of image encryption, it can be found that there exist many techniques that are capable of generating high degree of randomness in the cipher images and show good resilience against standard cryptanalysis attacks. However, the implementation strategies of these techniques consist of chaotic sequences, DNA computing, quantum CA, parallel computing, etc. each of which has enough resource consumption. Consequently, they are not suitable for the resource constrained devices used in perception layer of typical IoT applications such as healthcare, defense, surveillance and so forth. To overcome such limitations, in this paper, a lightweight image cipher, named as IEVCA has been proposed using 2-D VCA in Section 4. The proposed cipher is capable of resisting most of the standard cryptanalysis attacks and shows ample amount of randomness in cipher images. It can also be implemented in real time sensory devices in critical and sensitive IoT applications.

3 Preliminaries of 2-D cellular automata

CA are mathematical model through which behavior of a complex system can be explained through simple components acting together under a transformation rule. CA can be of many dimensions such as 1-D CA, 2-D CA, etc. Detailed description of 1D CA can be found in [40] and the references therein. In this work, 2-D Von-Neumann CA has been considered.

A 2VCA can have periodic-boundary or null-boundary considerations [26]. In periodic-boundary, the extreme boundary cells are considered as adjacent to each other while finding neighbor. On the contrary, in null-boundary CA, the extreme cells are considered to be connected to logic “zero”. Figure 1 shows the neighborhood considerations of a 2-D VCA.

The periodic-boundary and null-boundary conventions of Von-Neumann CA are shown in Fig. 2a and 2b respectively. The red colored cell (i, j) is the cell under consideration in Fig. 1 and Fig. 2. The positions of the neighbors are in North, South, East, West, and in the diagonal positions as shown in the figure. Each cell can have states or state-values as either ‘0’ or ‘1’. The state of a cell at position (i, j) at time t , can be changed depending

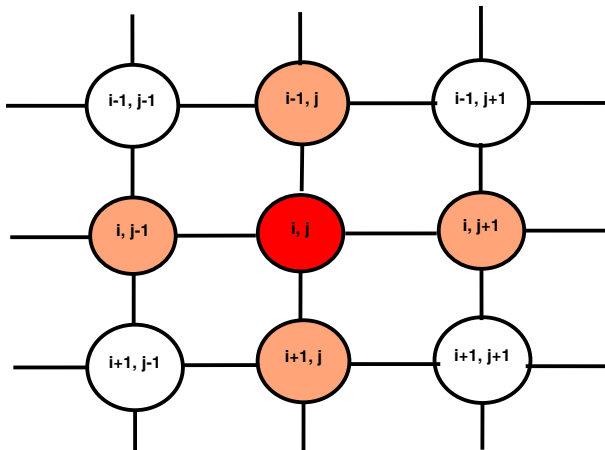


Fig. 1 Von Neumann neighborhood

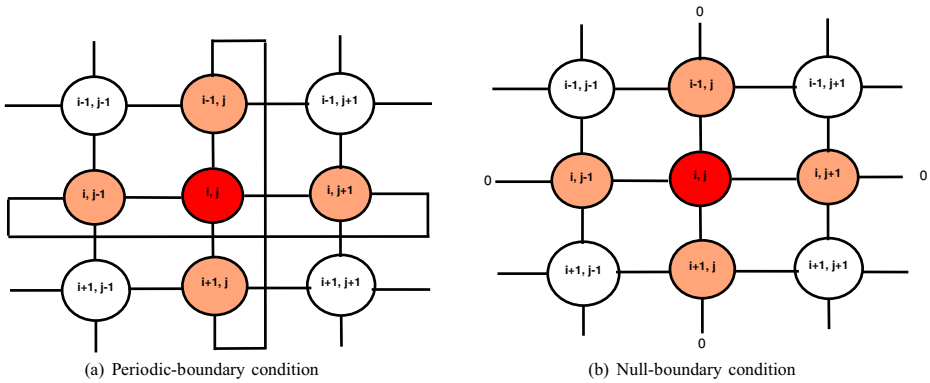


Fig. 2 Different types of Von Neumann CA and their representations

on the states of its neighbors at time $(t + 1)$ [55]. In this work, both periodic-boundary and null-boundary conventions of 2-D VCA are considered.

Figure 3 shows the block diagrammatic representation of 2-D periodic boundary Von Neumann CA, where the cells can be implemented through a D flip-flop and each input panel has 9 different lines connected through a XOR combination logic. The same for periodic boundary VCA can be designed if the horizontal and vertical boundary cells are connected to each other.

3.1 Mathematical model

2-D CA are an arrangement of $m \times n$ cells in matrix form, where state of each cell can be either '0' or '1'. A configuration means allocation of states to each and every cells of a 2-D CA. Each of such configurations can determine a next configuration through a local CA transformation rule. In other words, the state of a cell at time $(t + 1)$ depends only on the state of the cell and its neighbors (in any convention) at time t . Now, the state of $(i, j)^{th}$ cell

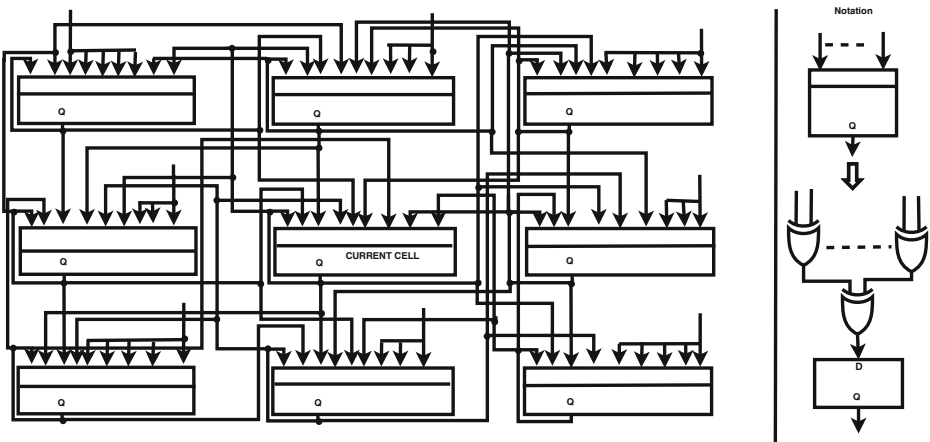


Fig. 3 Hardware implementation of 2-D Von Neumann CA with periodic boundary

at time $(t + 1)$ can be defined through (1). Table 1 lists all the abbreviations and notations used in the paper to describe the mathematical model.

$$s_{i,j}^{t+1} = w_1 \times s_{i,j}^t + w_2 \times s_{i,j+1}^t + w_4 \times s_{i+1,j}^t + w_8 \times s_{i,j-1}^t + w_{16} \times s_{i-1,j}^t \quad (1)$$

where, $w_i \in \{0, 1\}$. It further means that, for two-state 2-D CA, any value other than ‘0’ or ‘1’ will not be considered. The arithmetic summation of the cell values denoted in Table 2 denote the rule number over the field \mathbb{Z}^2 .

There are 5 fundamental rules: 1, 2, 4, 8 and 16. The position of the cells are illustrated in Table 2. As mentioned in the table, rule 1 signifies that the central cell has dependency on itself for next state transition. If the central cell depends on left cell for next state transition, it becomes rule 8. Similarly other rules can be constructed.

Now, in case of 2-D Von Neumann CA, highest possible rule number is 31, following the convention depicted in Table 2. The other rule numbers can be realized by taking arithmetic summation of the other rule numbers. For example, 2-D CA rule 23N $(1 + 2 + 4 + 16)$ signifies that the central cell changes its state depending on the states of its neighbors situated at the positions $(i, j - 1)$, $(i - 1, j)$, $(i, j + 1)$ as shown in Fig. 2b under null-boundary consideration. Whereas, rule 23P will do so under periodic-boundary consideration.

The behavior of such types of CA can be analyzed using mathematical model using matrices. Suppose, S_t is the matrix with initial configuration with a particular rule. The next state configuration can be obtained by suitable XOR operations of its respective neighbors under a specific rule. The global transitions can be expressed through the matrices M_1 and M_2 as given below:

$$M_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \quad M_2 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

The next states of the **Primary Rules** {1, 2, 4, 8, 16} can be computed as follows[27]:

$$\text{Rule 1 : } [S_{t+1}] = [S_t]; \quad \text{Rule 2 : } [S_{t+1}] = [S_t][M_2]; \quad \text{Rule 4 : } [S_{t+1}] = [M_1][S_t];$$

$$\text{Rule 8 : } [S_{t+1}] = [S_t][M_1]; \quad \text{Rule 16 : } [S_{t+1}] = [M_2][S_t];$$

Here, S_{t+1} denotes the state of the CA at time stamp $t + 1$. Detailed proof of the matrix representations of each primary rules over \mathbb{Z}^2 and \mathbb{Z}^3 can be found in [26].

Table 1 List of Abbreviation and Notations

Notations	Meaning
$s_{i,j}^{t+1}$	State of cell (i,j) at time t+1
$s_{i-1,j}^t$	State of cell (i-1,j) at time t
$s_{i+1,j}^{t+1}$	State of cell (i+1,j) at time t+1
23N	Rule number 23 with null boundary considerations
23P	Rule number 23 with periodic boundary considerations
$[T]_{i \times j}^k$	Characteristic matrix T of order k, having dimension $i \times j$
w_i	Weight of i^{th} positional value
2-D VCA	2 dimensional Von Neumann cellular automata
GCA	Group cellular automata
CARV	Cellular automata rule vector

Table 2 2-D Cellular Automata Rule Convention

	$2^4 (= 16)$	
$2^3 (= 8)$	$2^0 (= 1)$	$2^1 (= 2)$
	$2^2 (= 4)$	

For example, let us take $[S_t] = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. Now, $[S_{t+1}]$ can be computed using rule 23N as shown in (2). On the other hand using rule 23P, $[S_{t+1}]$ is computed as shown in Equation

$$[S_t] = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \xrightarrow{23N} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} = [S_{t+1}] \tag{2}$$

$$[S_t] = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \xrightarrow{23P} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = [S_{t+1}] \tag{3}$$

Here, we have applied same rule 23N or 23P to all cells of $[S_t]$ to obtain $[S_{t+1}]$. Hence, these are examples of uniform 2-D VCA. If different rules were applied in different cells, it would become hybrid 2-D VCA and in that case, the rules would be termed as 2-D CA rule vector, altogether.

4 IEVCA - the proposed technique

In a three layer IoT deployment scenario as depicted in Fig. 4, sensitive images are captured by camera sensors at the lowest layer, called physical layer or perception layer [32]. The captured images are then transmitted to the upper layer, called network layer through wi-fi or other public communication channels.

The network layer consists of gateway devices, router etc., with comparatively higher computing capabilities and storage than sensory devices deployed at perception layer. The nodes deployed in the network layer are also known as fog nodes that restricts huge network traffic towards application layer and thus increases the quality of service (QoS). In the application layer, there are high-end computers and servers for cloud storage. The users interact with the IoT applications through this layer and can obtain required data or analytical results available.

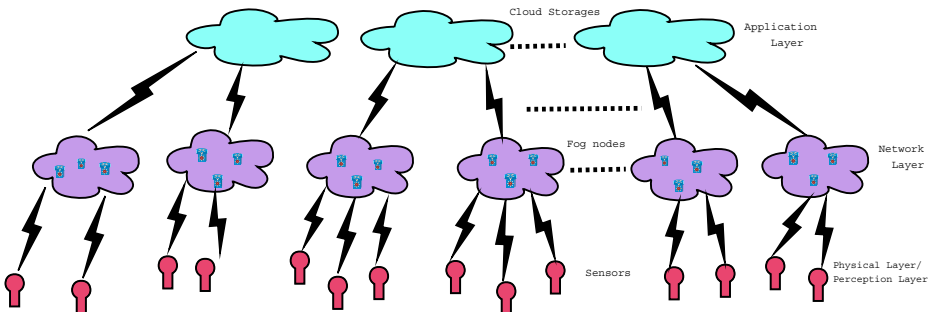


Fig. 4 Three layer IoT architecture

In this Section, at first we have generated the rule vectors and they are validated mathematically through state transition matrices in Section 4.1. After validation of the rule vectors, they are used in the IEVCA. Section 4.2 explains the detailed algorithm of rule generation by using both periodic and null boundary conditions. Section 4.3 and 4.4 describe encryption and decryption algorithm respectively in details.

4.1 2-D CA rule generation

The rule generation has been done by Algorithm 1. The detailed methods and technical specifications are explained in Section 4.2. Here, the illustrations regarding detailed operation of GCA is shown.

Any rule can be constructed through the primary rules, as mentioned earlier. For example, rule number 11P, 24P, 25P and 30P are constructed using the primary rules as shown in (4), (5), (6) and (7), respectively. Similarly, the other rules can be generated with periodic boundary or null boundary considerations.

$$\begin{aligned}
 \text{Rule } 11P &= \text{Rule } 1P + \text{Rule } 2P + \text{Rule } 8P \\
 \therefore [S_{t+1}] &= [S_t] + [S_t][M_2] + [S_t][M_1]
 \end{aligned}
 \tag{4}$$

$$\begin{aligned}
 \text{Rule } 24P &= \text{Rule } 8P + \text{Rule } 16P \\
 \therefore [S_{t+1}] &= [S_t][M_1] + [M_2][S_t]
 \end{aligned}
 \tag{5}$$

$$\begin{aligned}
 \text{Rule } 25P &= \text{Rule } 1P + \text{Rule } 8P + \text{Rule } 16P \\
 \therefore [S_{t+1}] &= [S_t] + [S_t][M_1] + [M_2][S_t]
 \end{aligned}
 \tag{6}$$

$$\begin{aligned}
 \text{Rule } 30P &= \text{Rule } 2P + \text{Rule } 4P + \text{Rule } 8P + \text{Rule } 16P \\
 \therefore [S_{t+1}] &= [S_t][M_2] + [M_1][S_t] + [S_t][M_1] + [M_2][S_t]
 \end{aligned}
 \tag{7}$$

A CA is uniform CA if all the cells transit under same rule; otherwise it is called hybrid CA. A CA is termed as group CA (GCA), if after certain transitions, say k the initial configuration is obtained back. In this case, k is the order of the group. Each CA has a characteristic matrix [40] through which it can be verified whether a CA is GCA. In case of GCA, (8) is satisfied.

$$[T]_{i \times j}^k = I_{i \times j}
 \tag{8}$$

A collection of different rules to form a GCA is called a GCA rule vector. For example, the rule vector given below is a 2-D GCA rule vector under null-boundary condition:

$$\text{Rule Vector} = \begin{pmatrix} 7N & 25N & 22N \\ 26N & 3N & 26N \\ 15N & 15N & 26N \end{pmatrix}$$

$$\begin{aligned}
 &\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
 &\rightarrow \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}
 \end{aligned}
 \tag{9}$$

The transition matrices for an initial configuration under above *RuleVector* are shown in (9). Here, the initial configuration has been re-generated after 6 rounds. So, it can be noted that for $k = 6$, $[T]_{3 \times 3}^6 = I_{3 \times 3}$.

Based on the dependency of central cell on the number of neighbors, the CA rules can be classified into many classes. The rule vectors used in this work are combination of rules that belong to different classes. This is done to ensure high degree of confusion and diffusion property in the cipher images.

In this work, we have generated Von Neumann, GCA rule vectors under both null-boundary and periodic-boundary conditions. Some of the null-boundary GCA rule vectors with value of $k = 6$ are given below:

$$\begin{aligned}
 CARV1 &= \begin{pmatrix} 31N & 11N & 22N \\ 26N & 3N & 26N \\ 15N & 15N & 26N \end{pmatrix} & CARV2 &= \begin{pmatrix} 15N & 9N & 5N \\ 26N & 2N & 26N \\ 15N & 15N & 26N \end{pmatrix} \\
 CARV3 &= \begin{pmatrix} 31N & 15N & 21N \\ 26N & 2N & 26N \\ 15N & 15N & 26N \end{pmatrix} & CARV4 &= \begin{pmatrix} 7N & 25N & 22N \\ 26N & 3N & 26N \\ 15N & 15N & 26N \end{pmatrix}
 \end{aligned}$$

Some of the periodic-boundary rule vectors with value of $k = 6$ used in this work are given below:

$$\begin{aligned}
 CARV1 &= \begin{pmatrix} 4P & 5P & 10P \\ 11P & 9P & 26P \\ 9P & 29P & 24P \end{pmatrix} & CARV2 &= \begin{pmatrix} 6P & 14P & 7P \\ 11P & 9P & 26P \\ 9P & 29P & 24P \end{pmatrix} \\
 CARV3 &= \begin{pmatrix} 10P & 5P & 6P \\ 11P & 9P & 26P \\ 9P & 29P & 24P \end{pmatrix} & CARV4 &= \begin{pmatrix} 23P & 30P & 12P \\ 11P & 9P & 26P \\ 9P & 29P & 24P \end{pmatrix}
 \end{aligned}$$

4.2 2-D VCA rule generation algorithm

There are many algorithms for providing high degree of security at the network layer and cloud layer, but communication from perception layer to network layer is the area of concern due to resource constraints and power limitations of the sensory devices. The traditional ciphers cannot be implemented at this layer.

Hence, the data sent from this layer are vulnerable to any adversary. To cope up with this, in this work a security model is proposed where the raw images sensed at physical layer are encrypted before sending them to the network layer, where they are decrypted. The block diagram of encryption and decryption processes are shown in Figs. 5 and 6, respectively.

The proposed algorithm is a symmetric key encryption technique, that performs encryption through pixel substitution by applying 2-D CA rule vectors having GCA property. Firstly, R, G and B channels are extracted from an input color image and are converted into binary matrices. Then, the rule scheduler randomly selects any three random CA rule vectors (CARVs), K_1 , K_2 and K_3 from 2D CARVList, where rule vectors are previously stored.

Then, these three channels are encrypted with K_1 , K_2 and K_3 randomly. The number of rounds to run for encryption is also selected at random by the encryption module. Finally, the encrypted matrices for R, G and B channels are combined to generate the cipher image for the input color image.

Subsequently, the cipher image is sent to the network layer where 2D CARVList and secret key, K are already stored. Here, it can be noted that the secret key used for encryption is $K = \langle K_1, K_2, K_3, r_{irr} \rangle$. We assume that this K can be shared between the physical layer and the network layer through some key management authority. Since, this issue falls beyond the scope of this work, we have made this assumption.

Algorithm 1 Rule Generator.**Require:** All of 511 2-D CA rules**Ensure:** 2D CARVList consisting of CA rule vector of length 9 and *iteration*.

```

1: for  $s \leftarrow 1$  to all possible 2D CARV do
2:    $count \leftarrow 0$ 
3:   while (status) do
4:     for  $k \leftarrow 1$  to 9 do
5:        $rule[k] \leftarrow \text{SelectRandom}(511 \text{ 2-D CA Rules})$       ▷ Periodic-boundary or
null-boundary rules are mutually exclusive
6:     end for
7:      $status \leftarrow \text{CheckRule}(rule)$       ▷ Returns true if rule is already verified
8:   end while
9:   for  $i \leftarrow 1$  to 31 do
10:     $previter \leftarrow -1, curiter \leftarrow -1$ 
11:     $data \leftarrow \text{DectoBinary}(i)$       ▷ Returns 2-D array of size  $3 \times 3$ 
12:     $copy \leftarrow data$ 
13:    for  $j \leftarrow 1$  to  $n$  do      ▷  $n$  is the number of iteration
14:       $copy \leftarrow \text{NullVon-Neumann}(rule, copy)$       ▷ Call Algorithm 4.2
(NullVon-Neumann) for null-boundary CARVs
15:       $matchc \leftarrow 0$ 
16:      for  $u \leftarrow 1$  to 3 do
17:        for  $v \leftarrow 1$  to 3 do
18:          if ( $copy[u][v] = data[u][v]$ ) then
19:             $matchc++$ 
20:          end if
21:        end for
22:      end for
23:      if ( $j < 16 \ \&\& \ match = 9$ ) then
24:         $curiter \leftarrow j$ 
25:      end if
26:    end for
27:    if ( $matchc = 9 \ \&\& \ i \neq 1$ ) then
28:      if ( $previter = curiter$ ) then
29:         $count++$ 
30:         $previter = curiter$ 
31:      else
32:        break
33:      end if
34:    else
35:      if ( $matchc = 9$ ) then
36:         $count++$ 
37:         $previter = curiter$ 
38:      end if
39:    end if
40:  end for
41:  if ( $count = 511$ ) then
42:    Add rule to 2-D CARVList
43:  end if
44: end for

```

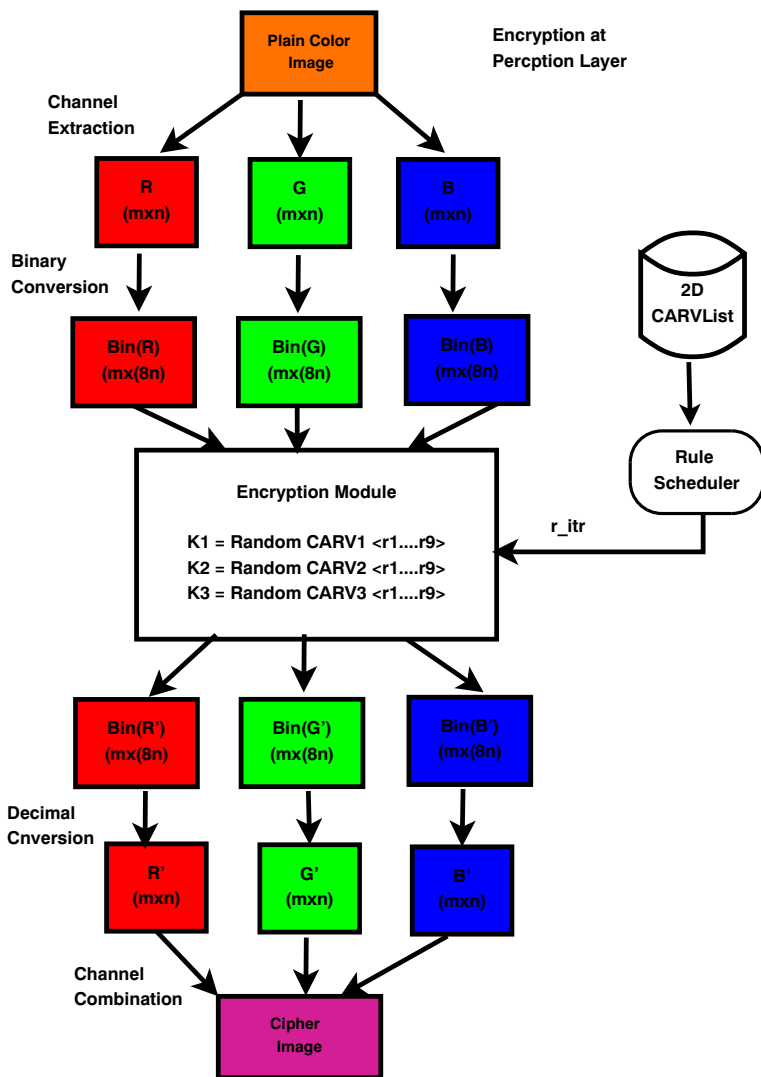


Fig. 5 Encryption process

Algorithm 4.2 is used to find 2-D CARVList based on periodic-boundary or null-boundary VCA. The all of 511 CA rules can be filled either by considering periodic-boundary or null-boundary exclusively at a time, without mixing both. Algorithm 4.2 considers all possible CA rules (for a 3×3 matrix, all possible rules are $(511)^9 = 2^{81}$ with repetitions) and produces a list containing valid GCA rule vectors along with the number of rounds, called *iteration*, the rule takes to cycle back to the original initial data. The validation of 2-D CA rules are shown in Section 4.1 with proper example. Step 14 of Algorithm 4.2 calls Algorithm 4.2 or 4.2 for selecting CARV considering null boundary or periodic boundary conditions respectively. These algorithms are considered on at a time. They should be mixed; otherwise CARV cannot be generated.

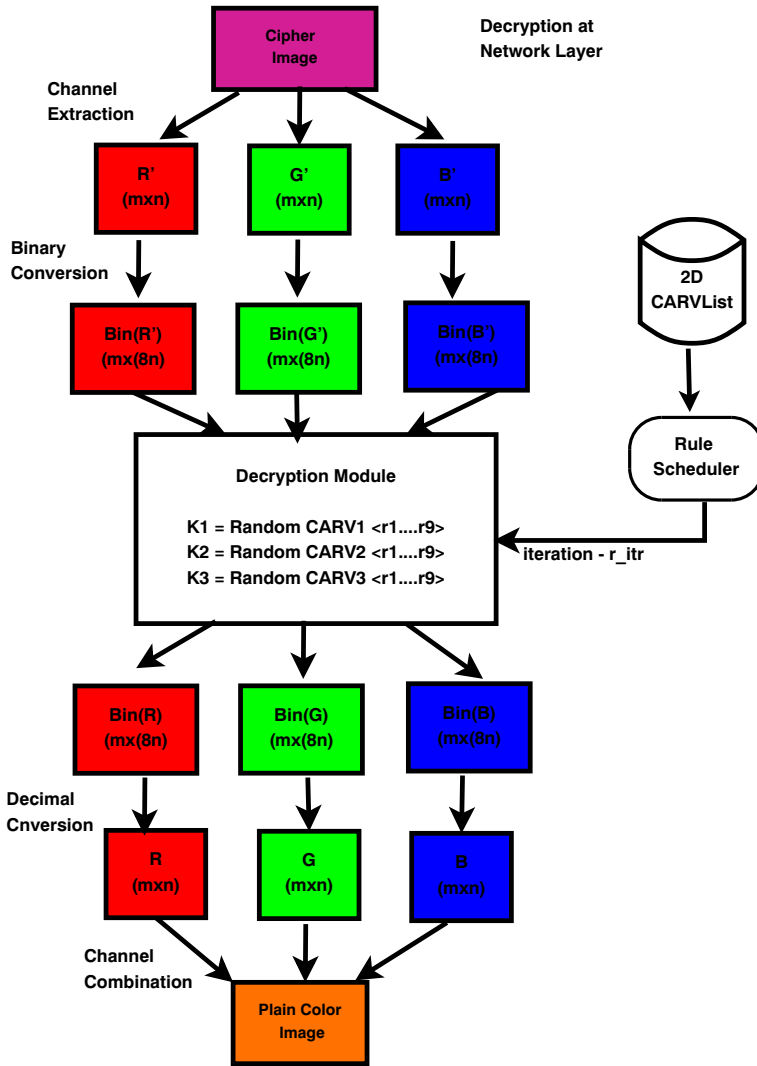


Fig. 6 Decryption process

Algorithm 4.2 takes a rule vector $\langle r_1, r_2, \dots, r_9 \rangle$ at random using “SelectRandom” method and verifies it for all data combinations possible for a 3×3 matrix to guarantee data independence, i.e., the initial data can be any random bits. The algorithm will consider any random 3×3 matrix and apply the randomly selected rule. Data independence signifies that the selected 2-D CARV will show GCA property for any 3×3 data. Step 9 iteratively selects a data value which is converted into 9 bits and then stored in a 3×3 matrix form, say D_t , where t is the time-stamp value. Step 9 iteratively selects a data value which is converted into 9 bits and then it is stored in a 3×3 matrix form, say D_t where t is the time-stamp value.

Algorithm 2 PeriodicVon-Neumann.**Require:** A matrix, *Data*, having dimension $m \times n$ **Ensure:** A matrix, *Data_applied*, having dimension $m \times n$ after applying periodic-boundary CARV

```

1: exdata ← expandData(data)           ▷ Returns matrix  $(m + 2) \times (n + 2)$  with
   periodic-boundary, as shown in Figure 2a
2: ruleMatrix ← getMatrix(rule) ▷ Converts rule vector  $\langle r_1, \dots, r_9 \rangle$  to a  $9 \times 9$  matrix
3: for  $i \leftarrow 2$  to  $m - 1$  do
4:   for  $j \leftarrow 2$  to  $n - 1$  do
5:     Initialize Von-Neumann neighbors using ruleMatrix[( $i - 2$ )  $\times$   $m + (j - 1)$ ] row
6:     XORdata ← fetchData(exdata, neighbor,  $i$ ,  $j$ ) ▷ Returns data to perform
   XOR depending on which neighbor is set in step 5
7:     temp[ $i - 1$ ][ $j - 1$ ] ← DoXOR(XORData)   ▷ Returns result XORData[ $i$ ] ^
   XORData[ $i + 1$ ]
8:   end for
9: end for
10: for  $i \leftarrow 1$  to  $m$  do
11:   for  $j \leftarrow 1$  to  $n$  do
12:     Data_applied[ $i$ ][ $j$ ] ← temp[ $i$ ][ $j$ ]
13:   end for
14: end for
15: return Data_applied

```

Algorithm 3 NullVon-Neumann.**Require:** A matrix, *Data*, having dimension $m \times n$ **Ensure:** A matrix, *Data_applied*, having dimension $m \times n$ after applying null-boundary CARV

```

1: exdata ← expandData(data)           ▷ Returns matrix  $(m + 2) \times (n + 2)$  with
   null-boundary as show in Figure 2b.
2: ruleMatrix ← getMatrix(rule) ▷ Converts rule vector  $\langle r_1, \dots, r_9 \rangle$  to a  $9 \times 9$  matrix
3: for  $i \leftarrow 2$  to  $m - 1$  do
4:   for  $j \leftarrow 2$  to  $n - 1$  do
5:     Initialize Von-Neumann neighbours using ruleMatrix[( $i - 2$ )  $\times$   $m + (j - 1)$ ]
   row
6:     XORdata ← fetchData(exdata, neighbor,  $i$ ,  $j$ ) ▷ Returns data to perform
   XOR depending on which neighbor is set in step 5
7:     temp[ $i - 1$ ][ $j - 1$ ] ← DoXOR(XORData)   ▷ Returns result XORData[ $i$ ] ^
   XORData[ $i + 1$ ]
8:   end for
9: end for
10: for  $i \leftarrow 1$  to  $m$  do
11:   for  $j \leftarrow 1$  to  $n$  do
12:     Data_applied[ $i$ ][ $j$ ] ← temp[ $i$ ][ $j$ ]
13:   end for
14: end for
15: return Data_applied

```

Step 13 of Algorithm 1 applies the selected periodic-boundary or null-boundary rule vector on this matrix by calling Algorithm 2 or Algorithm 3 respectively, that finds the neighbors to compute the next states of the matrix at time-stamp $t + 1$. Algorithm 2 takes an input matrix, $Data$ of size $m \times n$ and produces a matrix, $Data_applied$ of the same size, i.e., $m \times n$ by applying the selected periodic boundary cellular automata rule vector. Algorithm 3 does the same but it considers null boundary conditions while generating $Data_applied$.

As mentioned earlier, for a 2-D VCA, there are 9 possible neighbors. For implementation purpose, a neighbor is identified from the bit string, if the respective bit is '1' in the binary representation of the selected rule. Algorithm 2 and 3 initialize the neighbors as shown in Fig. 2a and b respectively. Upon finding the neighbors, "fetchData" method finds the corresponding state values (either '0' or '1'), on which EXOR operation is performed to compute the next state. This process is repeated for each cell of 3×3 matrix and then matrix D_{t+1} is obtained. D_{t+1} contains next state value after application of CARV. Finally, D_{t+1} is compared to D_0 and if $D_{t+1} = D_0$, the rule vector is tested for another 3×3 matrix having other set of data. This process guarantees data independence for the CARV. If the CARV under consideration is tested for all possible values and for each case $D_{t+1} = D_0$, the CARV is added to CARVList.

On the other hand, if $D_{t+1} \neq D_0$, step 13 continues until a match is found or the maximum number of *iteration* is reached. At last, if no match is found even after reaching maximum value of *iteration*, the randomly selected periodic-boundary or null-boundary CARV at step 5 of Algorithm 1 is discarded and a new CARV is selected. The CARV is added to the CARVList only if $D_{t+1} = D_0$ for all data values and for same *iteration*. Once the 2D CARVList is generated, it is stored in the ROM of sensor devices and shared with the network layer devices.

This rule generation process is executed only once before the encryption/ decryption processes commence. It can be noted that the periodic-boundary or null-boundary rules are considered separately at a time, i.e., the periodic-boundary and null-boundary 2-D CARVs are generated separately by running Algorithm 2 or Algorithm 3 respectively with Algorithm 1.

4.3 Image encryption algorithm

Algorithm 4 is used to encrypt an input color image I of size $m \times n$ using three 2D CA rule vectors (CARVs) generated by Algorithm 1 and r_{itr} selected by rule scheduler. Algorithm 4 generates the encrypted image, I_{enc} and the secret key, K to be used for decryption at receiver end. The decryption process is carried out through Algorithm 6. First, red, green and blue channels are extracted from I . Then each channel is transformed into binary format by converting each pixel value into 8 bits.

The binary form of each *Channel* is stored in matrix "BinImage" of dimension $p \times q$, where $p = m$ and $q = 8 \times n$. Then, "BinImage" is passed to "FindBlock" method that returns an integer, N_{Blocks} containing 3×3 blocks of "BinImage". Apparently, the value of N_{Blocks} is $\lceil \frac{p}{3} \rceil \times \lceil \frac{q}{3} \rceil$. These blocks will be actually substituted using CARVs $\{ r_1, r_2, \dots, r_9 \}$ to incorporate high level of confusion and diffusion property.

u blocks are combined together for encryption in step 5 of Algorithm 4. It increases the length of CARVs to $9 \times u$, increasing the key space to $2^{32 \times u}$. Rule Scheduler selects a random CARV from CARVList at step 6. Step 7 chooses randomly the number of rounds, r_{itr} for which selected K_i should be applied. In each round, for each block, Algorithm 5 is called. It takes "BinImage", u and the CARV to be used for encryption. It returns the next state values which are stored in the respective positions of "BinImage". After the completion

of step 16, we get encrypted “BinImage”. Upto this step, every operation is executed for each *Channel*, i.e., red, green and blue of *I*.

Algorithm 4 Encrypt Image.

Require: Plain Color Image, *I* of size $m \times n$, 2D CARVList.

Ensure: Encrypted Cipher Image, I_{enc} of size $m \times n$, K .

```

1: Channel  $\leftarrow$  ExtractChannels(I)
2: while (Channel) do
3:   BinImage  $\leftarrow$  getBinImage(Channel)  $\triangleright$  Dimension of BinImage is  $p \times q$ , where
    $p = m, q = n \times 8$ 
4:    $N_{Blocks} \leftarrow$  FindBlock(BinImage)  $\triangleright$  Finds number of blocks ( $= \lceil \frac{p}{3} \rceil \times \lceil \frac{q}{3} \rceil$ ) of size
    $3 \times 3$  from BinImage
5:    $u \leftarrow$  Number of blocks, from  $N_{Blocks}$  to be encrypted in one cycle
6:    $K_i \leftarrow$  RuleScheduler(2D CARVList,  $u$ )  $\triangleright$  Returns  $u$  number of  $K_i$ , where,
    $1 \leq i \leq 3$ 
7:    $r_{itr} \leftarrow$  SelectRandom(1 to iterations)  $\triangleright$  iterations is obtained from CARVList
8:   for  $j \leftarrow 0$  to  $N_{Blocks}$  with increment by  $u$  do
9:     for  $k \leftarrow j$  to  $j + u$  do
10:      for  $i \leftarrow 0$  to  $r_{itr}$  do
11:         $temp \leftarrow$  EncryptBlock(BinImage,  $k, K_i[k\%u]$ )
12:        set EncryptBlock(BinImage,  $temp, k$ )
13:      end for
14:    end for
15:  end for
16: end while
17: while (Channel) do
18:   EncryptedPixel  $\leftarrow$  getDecImage(BinImage)
19: end while
20:  $I_{enc} \leftarrow$  put EncryptedPixel[ $i$ ][ $j$ ] at position ( $p, q$ ) where  $p$  is even, when  $i < \lceil \frac{width}{2} \rceil$ 
   and odd when  $i \geq \lceil \frac{width}{2} \rceil$ . Similarly,  $q$  is even when  $j < \lceil \frac{height}{2} \rceil$  and odd when  $j \geq$ 
    $\lceil \frac{height}{2} \rceil$ . Here, width and height are width and height of the image
21:  $K \leftarrow \langle K_1, K_2, K_3, r_{itr} \rangle$ 

```

Algorithm 5 EncryptBlock.

Require: Binary Image of size $p \times q, u, K_i$

Ensure: Encrypted Block of Binary Image

```

1:  $row \leftarrow (\frac{u}{\lceil \frac{q}{3} \rceil}) \times 3$ 
2:  $column \leftarrow (u \% \lceil \frac{q}{3} \rceil) \times 3$ 
3: for  $i \leftarrow 0$  to 3 do
4:   for  $j \leftarrow 0$  to 3 do
5:      $VonNeumannN \leftarrow$  neighbor(rulevector[ $i \times 3 + j$ ])
6:      $XorData \leftarrow$  fetchData(BinImage,  $row + i, column + j, VonNeumannNN$ )
7:      $temp[i][j] \leftarrow$  DoXor(XorData)
8:   end for
9: end for

```

Then, for each *Channel*, the decimal values are obtained using “getDecImage” procedure and these are stored into the matrix, *EncryptedPixel*. One more shuffling is done at Step 20, to obtain encrypted cipher image, *I_{enc}*. Finally, *K* is generated by combining values of *K_i* of red, green and blue channels along with *r_{itr}*. Algorithm 5 is called each time when a block needs to be encrypted. It gets the required input from Step 11 of Algorithm 4. Finally, it enciphers the image block and then Algorithm 4 generates the cipher image after final round of shuffling, as mentioned in Step 20.

Algorithm 6 Decrypt Image.

Require: Encrypted Image, *I_{enc}* of size $m \times n$, *K*.

Ensure: Plain Color Image, *I* of size $m \times n$.

```

1:  $I_{enc} \leftarrow$  put  $C_{pixel}[i][j]$  at position  $(p, q)$  where  $p$  is even, when  $i < \lceil \frac{width}{2} \rceil$  and odd when  $i \geq \lceil \frac{width}{2} \rceil$ . Similarly,  $q$  is even when  $j < \lceil \frac{height}{2} \rceil$  and odd when  $j \geq \lceil \frac{height}{2} \rceil$ . Here, width and height are width and height of the image
2: Channel  $\leftarrow$  ExtractChannels(Ienc)
3: while (Channel) do
4:   BinImage  $\leftarrow$  getBinImage(Channel)  $\triangleright$  Dimension of BinImage is  $p \times q$ , where  $p = m, q = n \times 8$ 
5:    $N_{Blocks} \leftarrow$  FindBlock(BinImage)  $\triangleright$  Finds number of blocks ( $= \lceil \frac{p}{3} \rceil \times \lceil \frac{q}{3} \rceil$ ) of size  $3 \times 3$  from BinImage
6:    $u \leftarrow$  Number of blocks, from  $N_{Blocks}$  to be encrypted in one cycle
7:    $K_i \leftarrow$  RuleScheduler(2D CARVList, u)  $\triangleright$  Returns  $u$  number of  $K_i$ , where,  $1 \leq i \leq 3$ 
8:    $r_{itr} \leftarrow$  SelectRandom(1 to iterations)  $\triangleright$  iterations is obtained from CARVList
9:   for  $j \leftarrow 0$  to  $N_{Blocks}$  with increment by  $u$  do
10:    for  $k \leftarrow j$  to  $j + u$  do
11:      for  $i \leftarrow 0$  to  $r_{itr}$  do
12:         $temp \leftarrow$  EncryptBlock(BinImage,  $k, K_i[k \% u]$ )
13:        set EncryptBlock(BinImage,  $temp, k$ )
14:      end for
15:    end for
16:  end for
17: end while
18: while (Channel) do
19:   DecryptedPixel  $\leftarrow$  getDecImage(BinImage)
20: end while
21:  $I \leftarrow$  DecryptedPixels

```

4.4 Decryption algorithm

In network layer, the decryption process is done through Algorithm 6. Since this is a symmetric key algorithm, the secret key *K* is used for decryption of *I_{enc}*. The decryption process follows similar steps as that of encryption. At first, the pixels of *I_{enc}* are unshuffled and red, green and blue channels are extracted from the encrypted color image, *I_{enc}*. The unshuffling is done by following the reverse of Step 20 of Algorithm 4. The respective channels are converted into *BinImage*. The secret key, *K* is obtained from perception layer. The receiver applies the same CARVs for (*iteration* – *r_{itr}*) times. Furthermore, in Step 12 of

Algorithm 6, the EncryptBlock function is called to get decrypted blocks of cipher image. Subsequently, the decrypted red, green and blue channels are merged to get back the original color image. Thus, plain image is obtained at the network layer by the shared secret key, K .

5 Experimental results

To strengthen the claim of proposed IEVCA, we evaluated the performance of it using real simulation platform. Several analyses like correlation coefficient, histogram analysis, entropy, cipher image quality analysis, differential analysis, key space analysis, sensitivity analysis, etc. are performed for IEVCA as they are important evaluation parameters for any image encryption scheme. Since it is a lightweight encryption scheme, to assess the real time energy consumption and execution time, we have used Raspberry Pi 3, model B with Python language for implementing the algorithms. The Raspberry Pi 3 had WiFi and Bluetooth Low Energy (BLE) capabilities to increase the functionality and to power more powerful devices over the USB ports.

The detailed specifications of Raspberry Pi 3 are Quad Core 1.2GHz ARMv8 64bit CPU, 1GB RAM, WiFi and Bluetooth Low Energy (BLE) on board, 40-pin Extended GPIO, $4 \times$ USB 2 ports, 4 Pole stereo output and composite video port, Full size HDMI, CSI camera port for connecting a Raspberry Pi camera, DSI display port for connecting touchscreen display, Micro SD port for loading operating system and storing data. The part of connections with system and Raspberry Pi setup is shown in Fig. 7. The NIST and DIEHARD randomness tests and other performance comparisons, once the cipher image is generated, were carried out in a system with Intel (R) Core (TM) i5-8265U 1.60 GHz and 1.80 GHz CPU, 8 GB RAM, WINDOWS 10 Pro operating system.

Rule generation is carried out on the same machine, but having two NVIDIA GeForce GT 920M Graphics Processing Units (GPU). Operating system used is Ubuntu version 18.04.3. To speedup the process we have used Compute Unified Device Architecture (CUDA) compilation tools, release 9.1, V9.1.85. A total of 2^{27} threads are spanned to find GCA rules

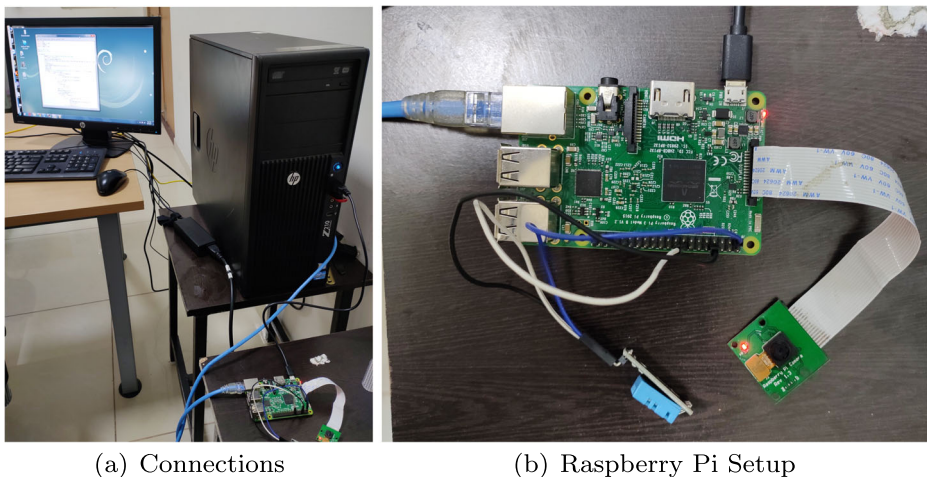


Fig. 7 Experimental setup for testing of proposed scheme

and each thread is responsible for verifying 2^{54} rules when considering Von-Neumann neighbors.

The plain color images used in this work are Baboon, Lena and Peppers as they contain essential features to be used for image processing experiments [41]. Figure 8 shows the plain, encrypted and decrypted images used in this work. Here, Periodic VCA and Null VCA represent the encryption method by using periodic-boundary VCA and null-boundary VCA, respectively. Cipher_PB and Cipher_NB denotes the cipher images obtained by Periodic VCA and Null VCA respectively, whereas, Dec_PB and Dec_NB represent the decrypted images through periodic VCA and Null VCA respectively. From Fig. 8, it can be observed that cipher images do not disclose any information regarding the plain images as per expectation.

6 Security analysis

Here, a detailed discussion about the robustness of the work based on various metrics is presented. This analysis confirms the effectiveness in preventing security attacks. IEVCA has been compared with state-of-the-art techniques to establish it as a better one. All these analysis confirms the feasibility of real-time implementation of the scheme in a constrained environment.

6.1 Analysis of key space

Key space represents the possible number of keys used for encryption algorithm. A higher key space makes the algorithm secure against brute force attacks making the key finding calculations computationally infeasible within a given interval of time. In this work, 3×3

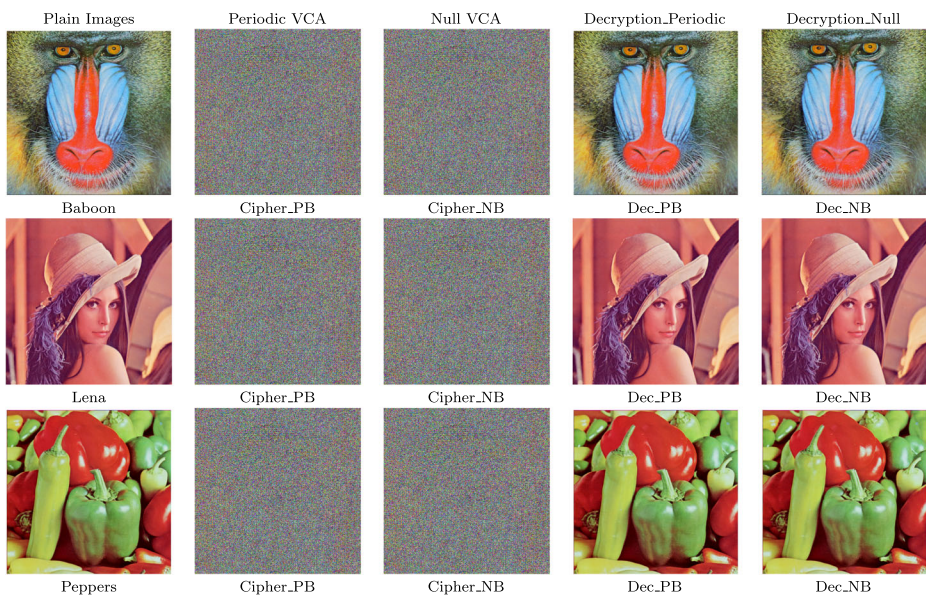


Fig. 8 Plain, encrypted and decrypted Images using periodic-boundary and null-boundary VCA

2-D CA with Von-Neumann neighborhood is considered where there are 9 neighbors of the central cell under consideration, including itself.

Each cell can be filled with the values ‘0’ or ‘1’. Hence, there are a total of 2^{81} of encryption keys for hybrid rule vectors that are used in this work. Now, suppose encryption is performed by considering P random blocks altogether, where P is a non zero integer. So, the total key-space becomes $2^{81 \times P}$, which is large enough to resist brute-force attacks with modern digital computers.

6.2 Resilience against cryptanalysis attacks

The proposed IEVCA is a symmetric key cipher that encrypts P number of blocks at a time, as mentioned in Section 4. While developing the proposed scheme, Kerckhoff’s principle [47, 62] has been obeyed. According to Kerckhoff’s principle, the encryption and decryption algorithms are known to public; only the encryption key is secret. In proposed scheme, every algorithm except the key is public. Below, we analyze the security of our proposed scheme against four classical attacks [62], provided the attacker knows the design and working principle of IEVCA .

1. **Ciphertext only attack:** In this kind of attack, the adversary has only a set of ciphertexts.
2. **Known plaintext attack:** Here, the adversary has a set of plaintexts and their corresponding ciphertexts.
3. **Chosen plaintext attack:** Here, we assume that the adversary has obtained ciphertexts for arbitrary plaintexts.
4. **Chosen ciphertext attack:** Here, the attacker has obtained the decrypted plaintexts of some chosen ciphertexts.

It can be easily understood from the description of the above mentioned classical attacks that the chosen plaintext attack is the most powerful attack among all. Hence, if a cryptosystem is resistant to this attack, it can prevent the other three types of attacks.

IEVCA is developed by using 2VCA that changes its configuration depending on 5 different neighbors dynamically. The rule vectors used exhibit GCA property. Now, in this scheme, three crucial components are chosen randomly - (1) the rule vectors K_1 , K_2 and K_3 from 2D CARVList (2) the particular K_i to be applied to each channel (R, G and B) and (3) the number of rounds, r_{itr} for which encryption method needs to be run. Consequently, the plain image to cipherimage mapping would be one-to-many, i.e., two same plain images can generate completely different cipher images depending on components of K . Even if same rule vectors of K is considered, then also the cipher images will be different based on r_{itr} . This has enabled IEVCA to achieve high degree of confusion and diffusion properties. Subsequently, it prevents the adversary from obtaining any meaningful information even if the adversary has the accessed the ciphertexts for some arbitrary plaintexts. Thus, IEVCA resists chosen plaintext, the most powerful cryptanalysis attack and hence it can resist the other three classical attacks.

6.3 Key sensitivity analysis

This analysis shows the sensitivity of the proposed scheme to the key used in the experiment. It indicates that a tiny alteration in private key can make the recovered image fully non-intelligible [61].

In other word, the modified key will generate a completely different cipher image. Figure 9 shows the decrypted image after a tiny alteration in the secret key.

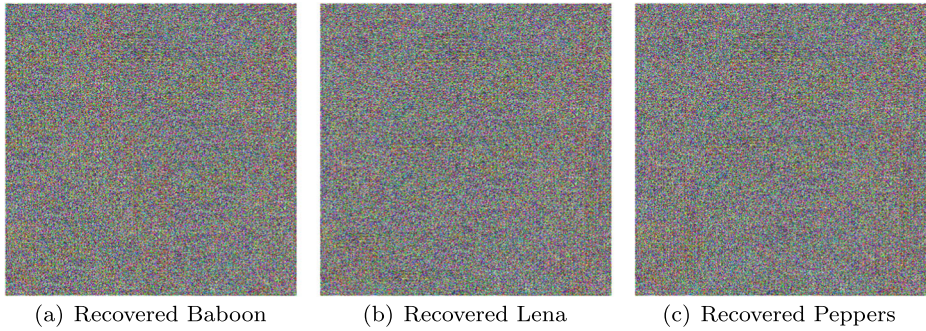


Fig. 9 Decryption of images with a small change in key

It can be observed that the scheme is highly sensitive to the secret key. IEVCA has been compared with the scheme proposed by Babaei et al. [1] and the result is shown in Table 3. It can be seen that the performance of the proposed scheme is better.

6.4 Information entropy analysis

Information entropy, α is one crucial metric for measuring the randomness in encrypted image. It can be calculated by (10) [66].

$$\alpha = \sum_{x=0}^{ij} S(T_x) \log_2 \frac{1}{S(T_x)} \tag{10}$$

Here, histogram count is denoted by $S(T_x)$. Total number of rows and columns of encrypted image are denoted by i and j respectively.

Table 4 provides the information entropy values of our scheme for red, green and blue channels of plain, encrypted and decrypted images. We have compared these values with the works of Babaei et al. [1], Jin et al. [23], Mondal et al. [39] and Fu et al. [18].

Table 5 shows the comparison data. It can be easily noticed from the table that IEVCA outperforms majority of the existing schemes and most of the attacks are infeasible in our case, as the entropy value is close to ideal values [22, 72].

6.5 Analysis of cipher image quality

Quality of cipher image generated after encryption is tested using two widely used metrics called, Mean Squared Error (MSE) and Peak Signal to Noise Ratio (PSNR). MSE and PSNR values have been computed using Equation (11) and (12) respectively.

$$MSE = \frac{\sum_{X,Y} [K_p(x, y) - K_c(x, y)]^2}{X \times Y} \tag{11}$$

Table 3 Key sensitivity test of proposed scheme (%)

Image	Babaei et al. [1]	Periodic VCA	Null VCA
Baboon	99.60	99.76	99.74
Lena	99.62	99.65	99.63
Peppers	99.66	99.71	99.68

Table 4 Entropy analysis of proposed scheme

	Baboon			Lena			Peppers		
	R	G	B	R	G	B	R	G	B
Plain Image	7.7155	7.4863	7.7638	7.2741	7.6758	7.3744	7.3244	7.4937	7.1487
Enc_Periodic_VCA	7.9992	7.9972	7.9984	7.9994	7.9996	7.9987	7.9986	7.9988	7.9992
Enc_Null_VCA	7.9939	7.9924	7.9938	7.9952	7.9873	7.9892	7.9967	7.9883	7.9791

$$PSNR = 10 \log_{10} \left(\frac{P^2}{MSE} \right) \tag{12}$$

Here, K_p and K_c denote original and encrypted images respectively. X, Y signify dimensions of the image in row and column. P is the value of maximum supported pixel of image matrix [41].

High value of MSE denotes that plain image is not understandable from cipher image. Similarly, PSNR value around 30 denotes presence of high degree of randomness in cipher image. A comparison has been performed with the encryption schemes of Jin et al. [23], Fu et al. [18], Tong et al. [53] and Mondal et al. [39]. The comparison result is listed in Table 6.

The values of MSE and PSNR between different plain images and decrypted images are provided in Table 7. This result is same for both periodic-boundary and null-boundary

Table 5 Comparison of information entropy values of the proposed scheme with existing techniques

Scheme	Images	Entropy-Value
Babaei et al. [1]	Baboon	7.9989
	Lena	7.9993
	Peppers	7.9993
Jin et al. [23]	Baboon	7.9934
	Lena	7.9927
	Peppers	7.9961
Mondal et al. [39]	Baboon	7.9993
	Lena	7.9993
	Peppers	7.9998
Fu et al. [18]	Baboon	7.9982
	Lena	7.9897
	Peppers	7.9562
Wang et al. [65]	Baboon	7.9969
	Lena	7.9970
	Peppers	–
Periodic VCA	Baboon	7.9996
	Lena	7.9997
	Peppers	7.9973
Null VCA	Baboon	7.9938
	Lena	7.9895
	Peppers	7.9884

Table 6 Comparison of image quality through MSE and PSNR between plain and cipher images

Image	Metrics	Baboon	Lena	Peppers
Jin et al. [23]	MSE	74.18	92.55	95.01
	PSNR	29.46	28.5	28.38
Fu et al. [18]	MSE	72.81	87.04	92.67
	PSNR	29.54	28.76	28.49
Tong et al. [53]	MSE	72.78	90.85	92.68
	PSNR	29.54	28.58	28.49
Modnal et al. [39]	MSE	72.73	90.73	92.54
	PSNR	29.54	28.58	28.5
Periodic VCA	MSE	82.38	82.56	93.37
	PSNR	28.95	28.95	28.42
Null VCA	MSE	82.48	82.87	93.37
	PSNR	28.96	28.94	28.41

conditions. The ‘0’ value of MSE and ∞ value of PSNR confirms that proposed IEVCA is a lossless image encryption technique.

6.6 Correlation coefficient analysis

Correlation coefficient is another statistical parameter of measuring the similarity of two images. It can be represented as β , to find out the relationship of two pixels of an image. a high value of β signifies the images are almost same, whereas, a very low value signifies that the images are completely different. The correlation coefficient β is computed through (13) [41].

$$\beta = \frac{\sum_i \sum_j (P_{ij} - \bar{P})(Q_{ij} - \bar{Q})}{\sqrt{(\sum_i \sum_j (P_{ij} - \bar{P})^2)(\sum_i \sum_j (Q_{ij} - \bar{Q})^2)}} \tag{13}$$

Here, \bar{P} and \bar{Q} denote the mean pixel value. P_{ij} represents the value at i^{th} row and j^{th} column.

Table 8 shows the values of correlation coefficient of plain image and cipher image and it also shows the comparison with the works done by Niyat et al. [43] and Babei et al. [1]. It can be observed that IEVCA has achieved better values of β .

Additionally, we have provided the plot of correlation coefficient values by taking 4000 pixel-value pairs at a time of plain image and cipher image. Figure 10 shows the correlation coefficient plots in horizontal, vertical and diagonal directions for the images of Baboon and Lena. The different plots of correlation coefficients of Peppers image are shown in Fig. 11.

Table 7 MSE and PSNR between plain and decrypted images

Image	MSE	PSNR
Baboon	0	∞
Lena	0	∞
Peppers	0	∞

Table 8 Comparison of correlation coefficient values of cipher images in horizontal, vertical and diagonal spectrum

Scheme	Images	Horizontal	Vertical	Diagonal
Niyat et al. [43]	Baboon	-0.0008	0.0002	-0.0006
	Lena	0.0022	0.0001	-0.0017
	Peppers	-0.0052	-0.0002	0.0005
Babaei et al. [1]	Baboon	-0.0014	-0.002	-0.0012
	Lena	0.0014	0.002	0.0012
	Peppers	-0.001	0.0011	0.0011
Periodic VCA	Baboon	-0.0021	-0.0015	-0.0057
	Lena	0.0010	-0.0030	-0.0011
	Peppers	0.0006	0.0007	0.0006
Null VCA	Baboon	-0.0024	0.0019	0.0023
	Lena	0.0053	0.0078	0.0042
	Peppers	0.0035	0.0018	0.0051

Here, Enc_Baboon_Periodic and Enc_Baboon_Null represent the encrypted images by periodic-boundary and null-boundary VCA, respectively. Similarly, the nomenclatures are used for other images.

It can be noted from Fig. 10 and Fig. 11 that the encrypted images have uniformly distributed pixel values, making them non-understandable for any attacker and it rightly justifies the efficacy of the proposed scheme.

6.7 Analysis of histogram

Histogram is a decisive feature of any digital color image as far as its encryption is concerned. It is the plot of tonal distribution of a color image.

It also represents graphically the distribution of number of pixel-values for each tone-value. The uniform and flat histogram indicates the presence of high degree of randomness in the pixel-values of cipher image. Table 9 shows the histogram of plain and encrypted cipher images of Baboon, Lena and Peppers respectively. Here, Enc_Baboon_Periodic and Enc_Baboon_Null signify the encrypted image using periodic-boundary VCA and null-boundary VCA, respectively. Similarly, the other images are denoted. The uniform and flat histogram of encrypted images represent the capability of generating random cipher image by IEVCA.

6.8 Robustness test against different levels of noise

The proposed scheme is tested with different noise level to check its robustness. At first “Salt and Pepper” noise is added to the plain images at different percentage, then these images are encrypted and sent to the receiver end.

After decryption, the original images are recovered at the network layer. Figure 12 and 13 show the plain images, encrypted images and recovered images using periodic-boundary VCA and null-boundary VCA, respectively when 1% noise level is considered. Subsequently, Fig. 14 and Fig. 15 show the respective images when 10% noise level is considered.

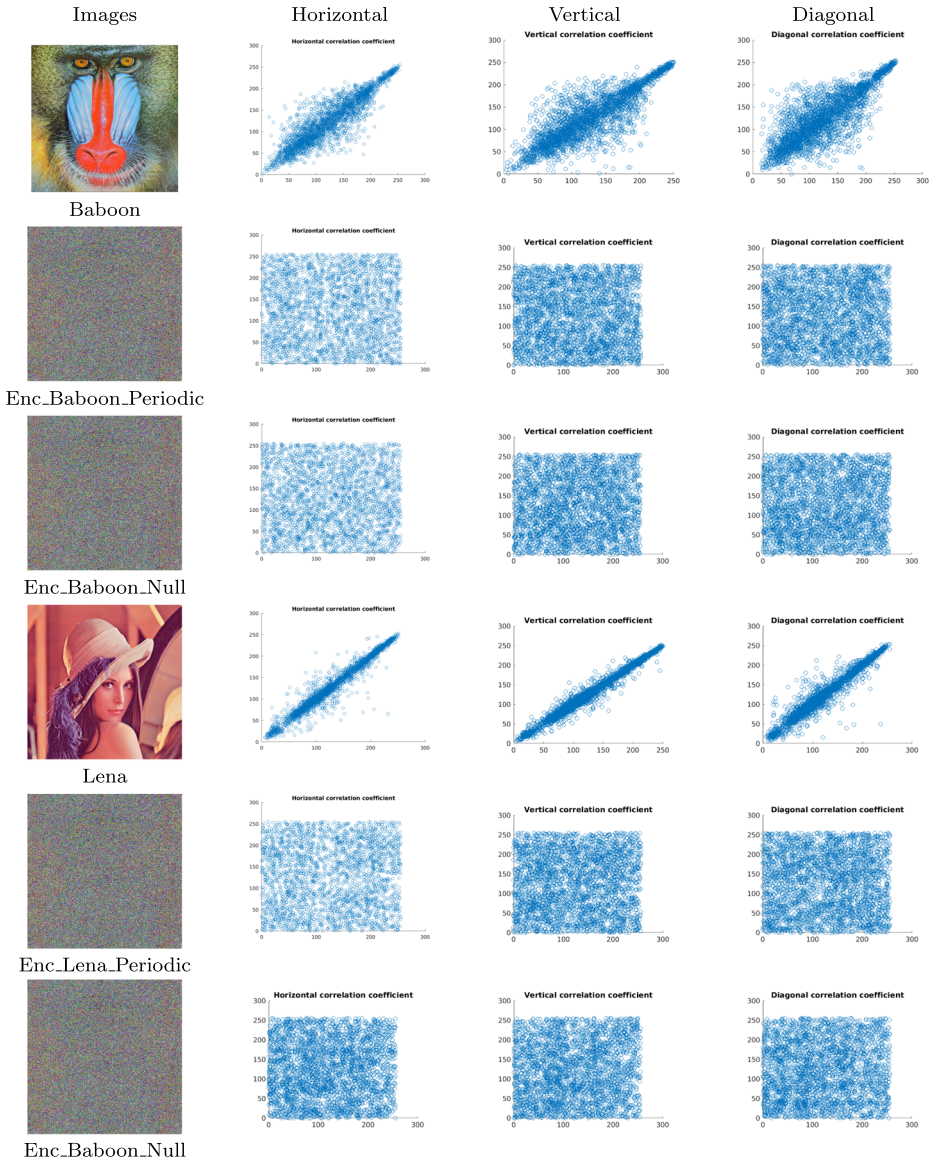


Fig. 10 Correlation coefficient of plain and encrypted baboon and lena images

The figures clearly show that the plain images can be easily recognized by observing the recovered images.

6.9 Differential analysis

An ideal characteristic of the encrypted image is its sensitivity to tiny changes in the plain image. An adversary tries to observe the change in cipher image when a tiny change occurs in the plain image. Then she tries to establish a relationship between plain image and the

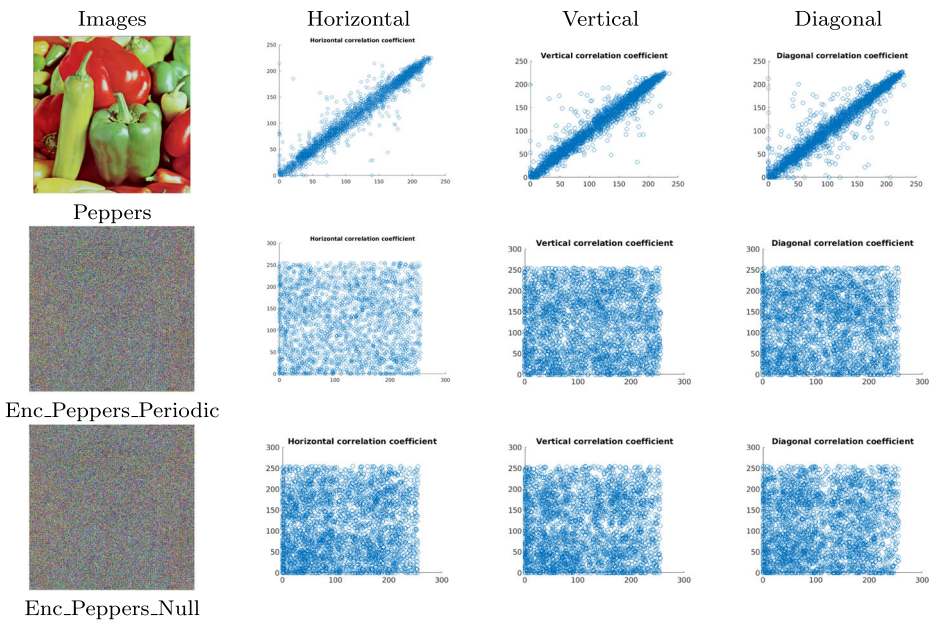


Fig. 11 Correlation coefficient of plain and encrypted pepper images

secret key. In fact, this process is called differential analysis where an adversary tries to find out the key. Number of Pixels Change Rate (NPCR) and Unified Average Changing Intensity (UACI) are two key criteria for testing the effect of a minor change in input image pixel on the cipher image. Higher values signifies the better efficiency of a image cipher [1]. NPCR and UACI values can be computed using (15) and (16) [39].

$$D(x, y) = \begin{cases} 0 & \text{if } D_1(x, y) = D_2(x, y) \\ 1 & \text{if } D_1(x, y) \neq D_2(x, y) \end{cases} \tag{14}$$

$$NPCR, N(D_1, D_2) = \sum \frac{D(x, y)}{M \times N} \times 100\% \tag{15}$$

$$UACI, U(D_1, D_2) = \sum_{x,y} \frac{|D_1(x, y) - D_2(x, y)|}{P \times M \times N} \times 100\% \tag{16}$$

Here, D_1 and D_2 are the encrypted images before and after one pixel change in plain image. $D_1(x, y)$ and $D_2(x, y)$ are positional pixel values of D_1 and D_2 respectively and are computed by (14). Here, M and N denotes dimension of the image and P is the highest value of a pixel in cipher image matrix.

The proposed scheme is compared with other existing schemes of Jin et al. [23], Fu et al. [18], Tong et al. [53], Nayak et al. [41] and Dong et al. [11]. The detailed result is listed in Table 10. It can be observed from the table that the values obtained are over and above the acceptable values [21] and outperforms majority of the existing schemes. Hence, the proposed scheme is capable to resist the differential cryptanalysis attacks.

Table 9 Histograms of different channels of plain and cipher images using periodic and null-boundary VCA

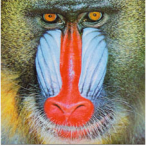
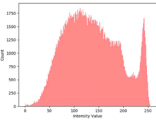
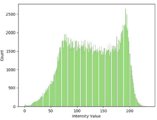
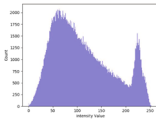
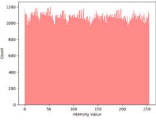
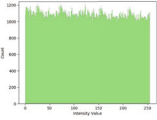
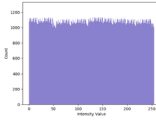
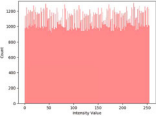
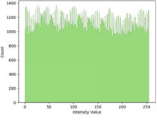
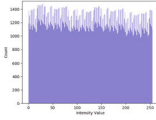
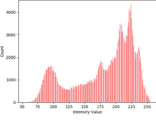
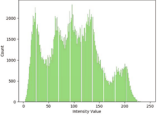
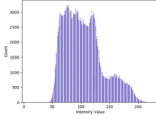
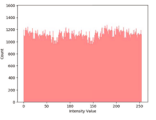
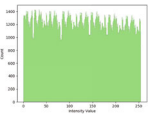
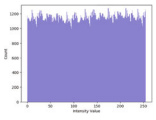
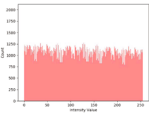
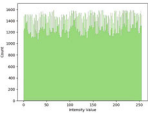
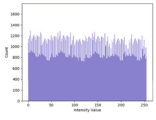
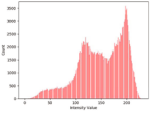
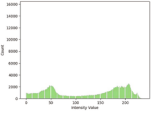
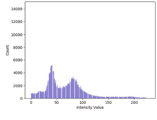
Images	Red Channel	Green Channel	Blue Channel
			
Baboon			
Enc_Baboon_Periodic			
Enc_Baboon_Null			
Lena			
Enc_Lena_Periodic			
Enc_Lena_Null			

Table 9 (continued)

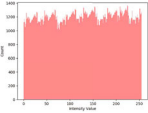
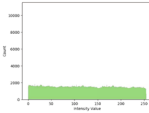
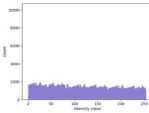
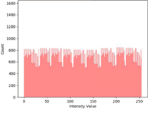
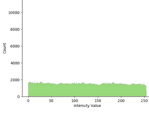
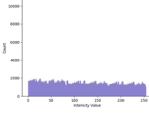
Images	Red Channel	Green Channel	Blue Channel
Peppers			
Enc_Peppers_Periodic			
Enc_Peppers_Null			



Fig. 12 Robustness test with 1% noise level for periodic-boundary VCA



Fig. 13 Robustness test with 1% noise level for null-boundary VCA

6.10 DIEHARD tests

The proposed scheme has undergone the statistical DIEHARD¹ randomness tests, introduced by G. Marsaglia (1995). The acceptable range of p-values is [0,1). The results are shown in Table 11. It can be seen that the p-values of our scheme is under the acceptable range, that denotes the cipher image is highly random.

6.11 NIST randomness tests

The randomness of the bit sequence generated in the cipher image is one of the key criteria to assess security of any image encryption scheme. There are many methods of testing randomness, but no algorithm has passed all of them. The most widely used set of randomness tests is SP800-22 published by the National Institute of Standards and Technology (NIST)².

¹<https://web.archive.org/web/20160125103112/http://stat.fsu.edu/pub/diehard/>

²<https://www.nist.gov/publications/statistical-test-suite-random-and-pseudorandom-number-generators-cryptographic>



Fig. 14 Robustness test with 10% noise level for periodic-boundary VCA

The statistical test suit (STS) consists of 15 different tests aiming to find non-randomness in a sequence. The tests are - I) Approximate Entropy, II) Block Frequency ($m=20000$), III) Cumulative Sums (Forward), IV) Frequency, V) Linear Complexity ($m=500$), VI) Longest Runs of 1s, VII) Maurer's Universal, VIII) Non-overlapping Templates, IX) Overlapping Templates, X) Random Excursions ($x=-1$), XI) Random Excursions Variant ($x=-1$), XII) Rank, XIII) Runs (Forward), XIV) Serial ($m=16$, $delsi2m$) and XV) Spectral DFT. NIST has provided two parameters of interpretation of the result: (1) proportion and (2) p-value. More details about these parameters can be found at [48]. In Table 12, the serial numbers of these tests are mentioned. The significance level, α is 0.01 for p-values obtained at each test. It means that if the $p\text{-value} \geq \alpha$, the sequence is random, otherwise it is non-random.

In our experiment, the value of $m = 100$ and hence acceptable proportion range is $[0.96015, 1.01985]$ [48]. The proposed scheme is compared with [48] in terms of obtained values of proportions and p-values. The results are listed in Table 12. Here, proportion and result are represented by Pr. and Re. It can be observed that the proportion value is under the acceptable range and the p-value is also > 0.001 that signify presence of high degree of randomness in cipher image.



Fig. 15 Robustness test with 10% noise level for null-boundary VCA

6.12 Comparison of runtime and energy consumption

In this section, a detailed comparison has been performed in terms of runtime and energy consumption for the proposed scheme. Here, symmetric block ciphers such as AES, DES, 3DES are considered along with lightweight stream cipher, Chacha for comparison purpose.

Table 10 Comparison of NPCR and UACI values with existing schemes

Image	Baboon		Lena		Peppers	
	NPCR	UACI	NPCR	UACI	NPCR	UACI
Jin et al. [23]	94.9703	31.5159	95.0157	33.7828	94.9356	31.2175
Fu et al. [18]	99.6025	31.0242	99.6928	33.0179	99.6743	32.8146
Wang et al. [60]	99.6048	33.5666	99.6567	33.4782	99.5865	33.5104
Tong et al. [53]	99.6259	32.3791	99.6296	33.7284	99.6201	32.9630
Periodic VCA	99.7412	33.4033	99.6323	33.3564	99.6123	33.4822
Null VCA	99.3645	32.1028	99.5268	32.7136	99.4327	33.1167

Table 11 DIEHARD test result

Scheme	Periodic VCA			Null VCA		
	512 × 512			512 × 512		
	p-value	Result	1024 × 1024	p-value	Result	1024 × 1024
Binary Rank 32*32	0.362348	✓	0.372728	0.345474	✓	0.386536
Binary Rank 6*8	0.924548	✓	0.978764	0.928363	✓	0.932652
Birthday Spacings	0.797924	✓	0.868537	0.783458	✓	0.846757
Overlapping 5-Permutation	0.855412	✓	0.924242	0.865327	✓	0.912293
OQSO	0.975122	✓	0.983788	0.984493	✓	0.924072
OPSO	0.985227	✓	0.966599	0.928292	✓	0.930753
Bitstream	0.162652	✓	0.282673	0.195302	✓	0.287487
DNA	0.479262	✓	0.799238	0.433483	✓	0.779438
Count-the-1's	0.8273503	✓	0.963064	0.7032652	✓	0.999153
Count-the-1's 2	0.9073439	✓	0.957748	0.9783588	✓	0.916208
Minimum Distance	0.8926093	✓	0.899136	0.898825	✓	0.843069
Squeeze	0.2581789	✓	0.318232	0.249229	✓	0.376212
Overlapping Sums	0.1772028	✓	0.277484	0.1527293	✓	0.275595
3D Spheres	0.1655452	✓	0.266882	0.186794	✓	0.286909
Parking Lot	0.687474	✓	0.764678	0.677312	✓	0.786676
Crap	0.886736	✓	0.976646	0.855582	✓	0.986321
Runs	0.886862	✓	0.8763732	0.823877	✓	0.855147

Table 12 NIST test result

Scheme	Ping et al.[48]			Periodic VCA			Null VCA					
	512×512	1024×1024	1024×1024	512×512	1024×1024	1024×1024	512×512	1024×1024	1024×1024			
Tests	p-value	Pr.	Re.	p-value	Pr.	Re.	p-value	Pr.	Re.	p-value	Pr.	Re.
I	0.4190	0.98	✓	0.8972	0.98	✓	0.8734	0.98	✓	0.8673	1.00	✓
II	0.9357	0.98	✓	0.9233	0.99	✓	0.9453	0.99	✓	0.8623	1.00	✓
III	0.1223	0.98	✓	0.3974	1.00	✓	0.3960	1.00	✓	0.5699	0.99	✓
IV	0.7597	1.00	✓	0.8645	1.00	✓	0.8573	1.00	✓	0.9184	1.00	✓
V	0.2368	0.99	✓	0.3535	0.99	✓	0.3732	1.00	✓	0.3856	1.00	✓
VI	0.9878	0.99	✓	0.9354	0.99	✓	0.9745	0.99	✓	0.9784	1.00	✓
VII	0.0757	1.00	✓	0.9218	0.98	✓	0.9756	0.98	✓	0.9367	0.98	✓
VIII	0.5544	0.99	✓	0.5534	0.99	✓	0.5846	0.99	✓	0.9944	1.00	✓
IX	0.9240	0.97	✓	0.9547	0.97	✓	0.9536	0.99	✓	0.6275	0.99	✓
X	0.6024	1.00	✓	0.5211	1.00	✓	0.4535	1.00	✓	0.4944	1.00	✓
XI	0.4685	1.00	✓	0.1344	0.99	✓	0.1242	0.98	✓	0.1365	0.98	✓
XII	0.2757	1.00	✓	0.1747	1.00	✓	0.3844	0.98	✓	0.3847	0.98	✓
XIII	0.9357	1.00	✓	0.9856	1.00	✓	0.8864	0.98	✓	0.8738	0.98	✓
XIV	0.8343	0.99	✓	0.9474	0.99	✓	0.9634	0.99	✓	0.8855	0.99	✓
XV	0.8977	0.99	✓	0.2733	0.99	✓	0.2821	0.99	✓	0.3563	0.99	✓

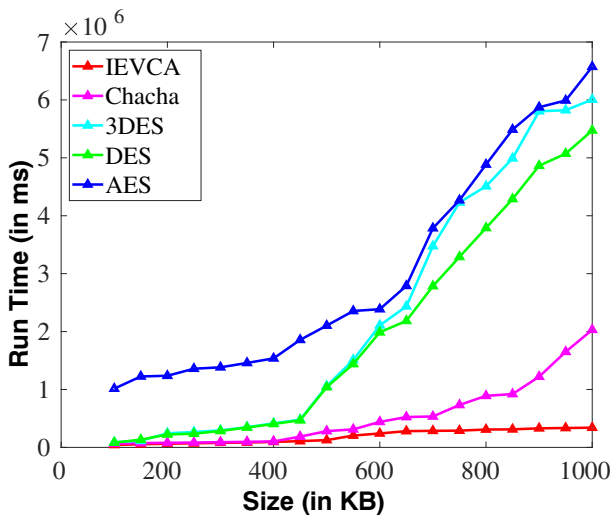


Fig. 16 Runtime comparison

6.12.1 Runtime comparison

IEVCA has been compared with AES, DES, 3-DES which are symmetric key block ciphers [17] and one lightweight stream cipher, called Chacha, from Sala20 family [3]. The result is shown in Fig. 16. Because of their complex fiestel structures, AES, DES and 3DES have higher runtime. Though Chacha is a lightweight stream cipher, but still it has higher runtime than IEVCA.

It can be observed from Fig. 16 that proposed scheme has the least runtime under similar implementation platform, making it suitable for implementation in realtime IoT applications. The the previous subsections, the proposed scheme has been compared with the existing schemes in terms of MSE, PNSR values for assessing the quality of encrypted image. It has been compared with respect to UACI and NPCR for evaluating its performance against differential attack. The entropy values, correlation coefficient comparisons confirms the security and resistance against cryptanalysis attacks like known plaintext, chosen plaintext, known ciphertext and chosen ciphertext. Hence, the proposed scheme manifests high randomness and resilience against cryptanalysis attacks and also achieves lesser runtime than other ciphers.

6.12.2 Energy consumption

The proposed method has been experimented using Raspberry Pi 3 model B, that has recommended power supply unit (PSU) current capacity of 2.5A. The typical bare board active current consumption was 400mA³. The supply voltage was provided to the board through microUSB power connector. The energy consumption was computed through the method described by [49]. According to their method, the energy consumed can be calculated by

³<https://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md>

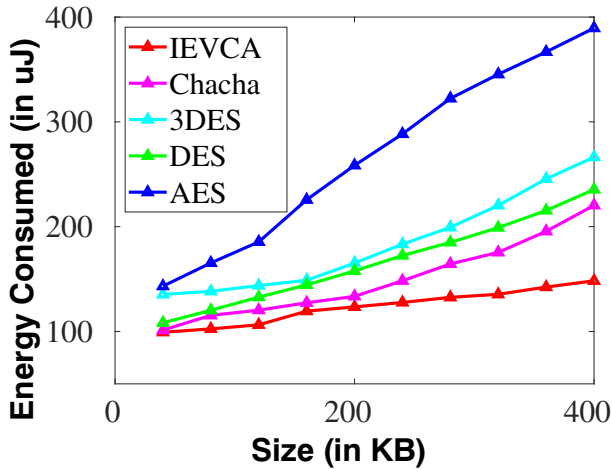


Fig. 17 Comparison of Energy Consumption

taking product of input supply voltage, power drawn by hardware and the average execution time of the algorithm. The supply voltage is 5V and it is constant for all the algorithms under consideration. We have compared the power consumption of IEVCA with the aforementioned algorithms. The result is shown in Fig. 17. It can be observed from Fig. 17 that as the size of the input increases, the energy consumption also increases. Among the aforementioned algorithms, our proposed scheme, IEVCA consumes the least amount of energy. It justifies one of the most important characteristics of lightweight algorithm.

7 Conclusion and future works

Proposed work presents an image encryption technique called, IEVCA using 2-D Von-Neumann Cellular Automata. The lightweight feature of IEVCA makes its runtime faster as compared to its existing counterparts. Hence, it is more appropriate for the IoT applications developed for sensitive fields like defense, healthcare, etc., which require encryption in the perception layer. The IoT sensory devices could be able to run the encryption routine with very less resource consumption. Many security and performance analyses are performed on IEVCA which confirm its robustness and resistance against various security attacks. Additionally, NIST and DIEHARD statistical test suites are used to check the randomness of the cipher image generated and we found that the cipher image passed all the tests.

In future, an efficient version of IEVCA may be planned with the use of cache efficient algorithms. The locality of reference feature of cache will be utilized for faster fetching of the rule vectors, as a result less resources of the IoT devices will be used. Further, IEVCA can be used along with trust management [37] and key management [31] to deploy secure applications. IEMCA will be complete when the secret key sharing between sender and receiver will be taken care of and will be optimized according to the requirements of the IoT deployment. The proposed scheme can be used to secure ambient agents [5, 36], as they also require similar type of security requirements.

References

1. Babaei A, Motameni H, Enayatifar R (2020) A new permutation-diffusion-based image encryption technique using cellular automata and dna sequence. *Optik* 203:164000
2. Bakhshandeh A, Eslami Z (2013) An authenticated image encryption scheme based on chaotic maps and memory cellular automata. *Opt Lasers Eng* 51(6):665–673
3. Bernstein DJ (2008) Chacha, a variant of salsa20. In: Workshop Record of SASC, vol 8, pp 3–5
4. Beyer WA, Sellers PH, Waterman MS (1985) Stanislaw m. ulam's contributions to theoretical theory. *Lett Math Phys* 10(2):231–242
5. Bouchemal N, Maamri R, Chihoub M (2013) Securing ambient agents groups by using verification, judgment and surveillance. *International Journal of Ambient Computing and Intelligence (IJACI)* 5(3):44–60
6. Chattopadhyay P, Choudhury PP, Dihidar K (1999) Characterisation of a particular hybrid transformation of two-dimensional cellular automata. *Computers & Mathematics with Applications* 38(5-6):207–216
7. Chen R-J, Lai Y-T, Lai J-L (2006) Architecture design and vlsi hardware implementation of image encryption/decryption system using re-configurable 2d von neumann cellular automata. In: 2006 IEEE International Symposium on Circuits and Systems, IEEE, pp 4–pp
8. Choudhury PP, Nayak BK, Sahoo S, Rath SP (2008) Theory and applications of two-dimensional, null-boundary, nine-neighborhood, cellular automata linear rules. [arXiv:0804.2346](https://arxiv.org/abs/0804.2346)
9. Diffie W, Hellman M (1976) New directions in cryptography. *IEEE transactions on Information Theory* 22(6):644–654
10. Dihidar K, Choudhury PP (2004) Matrix algebraic formulae concerning some exceptional rules of two-dimensional cellular automata. *Inf Sci* 165(1-2):91–101
11. Dong C (2014) Color image encryption using one-time keys and coupled chaotic systems. *Signal Processing Image Communications* 29(5):628–640
12. Enayatifar R, Guimarães FG, Siarry P (2019) Index-based permutation-diffusion in multiple-image encryption using dna sequence. *Opt Lasers Eng* 115:131–140
13. Enayatifar R, Sadaei HJ, Abdullah AH, Lee M, Isnin IF (2015) A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata. *Opt Lasers Eng* 71:33–41
14. Faheem M, Butt RA, Raza B, Ashraf MW, Ngadi MA, Gungor VC (2019) A multi-channel distributed routing scheme for smart grid real-time critical event monitoring applications in the perspective of industry 4.0. *International Journal of Ad Hoc and Ubiquitous Computing* 32(4):236–256
15. Faheem M, Gungor VC (2018) Mqrp: Mobile sinks-based qos-aware data gathering protocol for wireless sensor networks-based smart grid applications in the context of industry 4.0-based on internet of things. *Futur Gener Comput Syst* 82:358–374
16. Faheem M, Shah SBH, Butt RA, Raza B, Anwar M, Ashraf MW, Ngadi MA, Gungor VC, communication Smartgrid (2018) Information technologies in the perspective of industry 4.0 Opportunities and challenges. *Computer Science Review* 30:1–30
17. Feistel H (1973) Cryptography and computer privacy. *Scientific american* 228(5):15–23
18. Fu C, Lin B, Miao Y, Liu X, Chen J (2011) A novel chaos-based bit-level permutation scheme for digital image encryption. *Optics communications* 284(23):5415–5423
19. Granjal J, Monteiro E, Silva JS (2015) Security for the internet of things: a survey of existing protocols and open research issues. *IEEE Communications Surveys & Tutorials* 17(3):1294–1312
20. Heuer J, Hund J, Pfaff O (2015) Toward the web of things: Applying web technologies to the physical world. *Computer* 48(5):34–42
21. Hua Z, Zhou Y (2016) Image encryption using 2d logistic-adjusted-sine map. *Inf Sci* 339:237–253
22. Hua Z, Zhou B, Zhou Y (2018) Sine chaotification model for enhancing chaos and its hardware implementation. *IEEE Trans Ind Electron* 66(2):1273–1284
23. Jin J (2012) An image encryption based on elementary cellular automata. *Opt Lasers Eng* 50(12):1836–1843
24. Karmakar J, Nandi D, Mandal M (2020) A novel hyper-chaotic image encryption with sparse-representation based compression, multimedia tools and applications, 1–24
25. Kaur M, Kumar V (2018) A comprehensive review on image encryption techniques. *Archives of Computational Methods in Engineering* 27:1–29
26. Khan AR, Choudhury PP, Dihidar K, Mitra S, Sarkar P (1997) Vlsi architecture of a cellular automata machine. *Computers & Mathematics with Applications* 33(5):79–94
27. Khan AR, Choudhury PP, Dihidar K, Verma R (1999) Text compression using two-dimensional cellular automata. *Computers & Mathematics with Applications* 37(6):115–127

28. Khan MA, Salah K (2018) Iot security: review, blockchain solutions, and open challenges. *Futur Gener Comput Syst* 82:395–411
29. Khashan OA, AlShaikh M (2020) Edge-based lightweight selective encryption scheme for digital medical images, *multimedia tools and applications*, 1–20
30. Khedmati Y, Parvaz R, Behroo Y (2020) 2D hybrid chaos map for image security transform based on framelet and cellular automata. *Inf Sci* 512:855–879
31. Kimbahune VV, Deshpande AV, Mahalle PN (2017) Lightweight key management for adaptive addressing in next generation internet. *International Journal of Ambient Computing and Intelligence (IJACI)* 8(1):50–69
32. Lin J, Yu W, Zhang N, Yang X, Zhang H, Zhao W (2017) A survey on internet of things: architecture, enabling technologies, security and privacy, and applications. *IEEE Internet Things J.* 4(5):1125–1142
33. Liu H, Wang X (2010) Color image encryption based on one-time keys and robust chaotic maps. *Computers & Mathematics with Applications* 59(10):3320–3327
34. Liu H, Wang X (2011) Color image encryption using spatial bit-level permutation and high-dimension chaotic system. *Opt Commun* 284(16-17):3895–3903
35. Liu H, Wang X et al (2012) Image encryption using dna complementary rule and chaotic maps. *Appl Soft Comput* 12(5):1457–1466
36. Makri E, ten Brinke J, Evers R, Man P, Olthof H (2018) Privacy-friendly wi-fi-based occupancy estimation with minimal resources. *International Journal of Ambient Computing and Intelligence (IJACI)* 9(4):34–51
37. Mhetre NA, Deshpande AV, Mahalle PN (2016) Trust management model based on fuzzy approach for ubiquitous computing. *International Journal of Ambient Computing and Intelligence (IJACI)* 7(2):33–46
38. Mohamed FK (2014) A parallel block-based encryption schema for digital images using reversible cellular automata. *Engineering Science and Technology, an International Journal* 17(2):85–94
39. Mondal B, Singh S, Kumar P (2019) A secure image encryption scheme based on cellular automata and chaotic skew tent map. *Journal of information security and applications* 45:117–130
40. Nandi S, Kar B, Chaudhuri PP (1994) Theory and applications of cellular automata in cryptography. *IEEE Transactions on computers* 43(12):1346–1357
41. Nayak P, Nayak SK, Das S (2018) A secure and efficient color image encryption scheme based on two chaotic systems and advanced encryption standard. In: 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, pp 412–418
42. Neshenko N, Bou-Harb E, Crichigno J, Kaddoum G, Ghani N (2019) Demystifying iot security: an exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations. *IEEE Communications Surveys & Tutorials* 21(3):2702–2733
43. Niyat AY, Moattar MH, Torshiz MN (2017) Color image encryption based on hybrid hyper-chaotic system and cellular automata. *Opt Lasers Eng* 90:225–237
44. Noura H, Chehab A, Noura M, Couturier R, Mansour MM (2019) Lightweight, dynamic and efficient image encryption scheme. *Multimedia Tools and Applications* 78(12):16527–16561
45. Omoniwa B, Hussain R, Javed MA, Bouk SH, Malik SA (2018) Fog/edge computing-based iot (feciot): architecture, applications, and research issues. *IEEE Internet Things J.* 6(3):4118–4149
46. Panarello A, Tapas N, Merlino G, Longo F, Puliafito A (2018) Blockchain and iot integration: a systematic survey. *Sensors* 18(8):2575
47. Pareek N, Patidar V, Sud K (2005) Cryptography using multiple one-dimensional chaotic maps. *Commun Nonlinear Sci Numer Simul* 10(7):715–723
48. Ping P, Xu F, Wang Z-J (2014) Image encryption based on non-affine and balanced cellular automata. *Signal Process* 105:419–429
49. Prasithsangaree P, Krishnamurthy P (2003) Analysis of energy consumption of rc4 and aes algorithms in wireless lans. In: GLOBECOM'03. IEEE Global Telecommunications Conference (IEEE Cat. No. 03CH37489), iee, vol 3, pp 1445–1449
50. Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. *Commun ACM* 21(2):120–126
51. Sanchez-Avila C, Sanchez-Reillo R (2001) The rijndael block cipher (aes proposal): a comparison with des. In: Proceedings IEEE 35th Annual 2001 International Carnahan Conference on Security Technology (Cat. No. 01CH37186), IEEE, pp 229–234
52. Siap I, Akin H, Sah F (2011) Characterization of two-dimensional cellular automata over ternary fields. *J Frankl Inst* 348(7):1258–1275
53. Tong X-J, Wang Z, Zhang M, Liu Y (2013) A new algorithm of the combination of image compression and encryption technology based on cross chaotic map. *Nonlinear Dynamics* 72(1-2):229–241
54. Torres-Huitzil C (2013), IEEE, LASCAS

55. Uguz S, Sahin U, Sahin F (2013) Uniform cellular automata linear rules for edge detection. In: 2013 IEEE International Conference on Systems, Man, and Cybernetics, IEEE, pp 2945–2950
56. Von Neumann J, Burks AW et al (1966) Theory of self-reproducing automata. *IEEE Transactions on Neural Networks* 5(1):3–14
57. Wang X, Feng L, Zhao H (2019) Fast image encryption algorithm based on parallel computing system. *Inf Sci* 486:340–358
58. Wang X, Gao S (2020) Image encryption algorithm based on the matrix semi-tensor product with a compound secret key produced by a boolean network. *Inf Sci* 539:195–214
59. Wang X, Gao S (2020) Image encryption algorithm for synchronously updating boolean networks based on matrix semi-tensor product theory. *Inf Sci* 507:16–36
60. Wang X, Liu L, Zhang Y (2015) A novel chaotic block image encryption algorithm based on dynamic random growth technique. *Opt Lasers Eng* 66:10–18
61. Wang X, Luan D (2013) A novel image encryption algorithm using chaos and reversible cellular automata. *Commun Nonlinear Sci Numer Simul* 18(11):3075–3085
62. Wang X, Teng L, Qin X (2012) A novel colour image encryption algorithm based on chaos. *Signal Process* 92(4):1101–1108
63. Wang H, Xiao D, Li M, Xiang Y, Li X (2019) A visually secure image encryption scheme based on parallel compressive sensing. *Signal Process* 155:218–232
64. Wang X-Y, Yang L, Liu R, Kadir A (2010) A chaotic image encryption algorithm based on perceptron model. *Nonlinear Dynamics* 62(3):615–621
65. Wang X-Y, Zhang Y-Q, Bao X-M (2015) A novel chaotic image encryption scheme using dna sequence operations. *Opt Lasers Eng* 73:53–61
66. Wu X, Wang K, Wang X, Kan H (2017) Lossless chaotic color image cryptosystem based on dna encryption and entropy. *Nonlinear Dynamics* 90(2):855–875
67. Xu Y, Helal A (2015) Scalable cloud–sensor architecture for the internet of things. *IEEE Internet Things J*. 3(3):285–298
68. Yang Y-G, Tian J, Lei H, Zhou Y-H, Shi W-M (2016) Novel quantum image encryption using one-dimensional quantum cellular automata. *Inf Sci* 345:257–270
69. Yang Y, Wu L, Yin G, Li L, Zhao H (2017) A survey on security and privacy issues in internet-of-things. *IEEE Internet Things J*. 4(5):1250–1258
70. Zhang Y-Q, He Y, Li P, Wang X-Y (2020) A new color image encryption scheme based on 2dnclml system and genetic operations. *Opt Lasers Eng* 128:106040
71. Zhou J, Cao Z, Dong X, Vasilakos AV (2017) Security and privacy for cloud-based iot: Challenges. *IEEE Commun Mag* 55(1):26–33
72. Zhu C (2012) A novel image encryption scheme based on improved hyperchaotic sequences. *Optics communications* 285(1):29–37

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Satyabrata Roy¹ · Manu Shrivastava¹  · Chirag Vinodkumar Pandey¹ · Sanjeet Kumar Nayak² · Umashankar Rawat¹

Satyabrata Roy
satyabrata.roy@jaipur.manipal.edu

Chirag Vinodkumar Pandey
chiragpandey15@gmail.com

Sanjeet Kumar Nayak
sanjeet.nayak@bennett.edu.in

Umashankar Rawat
umashankar.rawat@jaipur.manipal.edu

¹ Department of Computer Science and Engineering, Manipal University Jaipur, Jaipur, India

² Department of Computer Science Engineering, Bennett University, Greater Noida, India