International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

DESIGN AND DEVELOPMENT OF A LIGHTWEIGHT ACCESS CONTROL FRAMEWORK FOR IOT DEVICE SECURITY

[1]Sayali C. Renuse, [2]Laxmi A. Bewoor, [3]Parikshit N. Mahalle

[1]Research Scholar, Department of Computer Engineering, Vishwakarma institute of information Technology pune-48, Savitribai phule Pune university (SPPU), Pune, Maharashtra, India. sayali.221p0081@viit.ac.in

[2]Research Supervisor, Associate Professor, Department of Computer Engineering, Vishwakarma institute of Technology pune-48, Savitribai phule pune university (SPPU), Pune, Maharashtra, India. laxmi.bewoor@vit.edu

[3]Professor, Department of Artificial Intelligence and Data Science,Vishwakarma institute of Technology pune-48, Savitribai phule pune university (SPPU), Pune.Maharashtra, India. aalborg.pnm@gmail.com

Abstract

There are a lot of Internet of Things (IoT) gadgets in important areas that need strong security systems to protect against possible risks. Based on the idea of least privilege, this paper describes a lightweight access control system that was made for IoT settings. For safe communication between machines, the suggested system uses a Capability-Based Access Control (CapBAC) model that has been improved with Advanced Encryption Standard (AES) and Reed-Solomon error correction. Our method includes making JSON-encoded capability tokens that describe device rights and have an expiration date. By hashing these tokens, symmetric AES keys are made, which protects the security and privacy of the data while it is being sent. Simulations of communication mistakes are used to test the system and see how well Reed-Solomon decoding and AES decryption work. Our method not only protects the privacy and security of data, but it also has a small operating size that makes it ideal for IoT devices with limited space. CapBAC's focus on both mistake repair and encryption shows that the framework has the ability to provide strong security in IoT networks that are always changing, where other security methods might not work.

Keywords: IoT Security, Access Control, Encryption, Error Correction, Capability-Based Access Control, Lightweight Security Framework

1. Introduction

In recent world, the Internet of Things (IoT) has become a major force in technology. It adds intelligence to everyday items and lets them talk and connect with each other through current Internet infrastructure. The fast spread of IoT devices in many areas, such as healthcare, home automation, industrial control, and transportation, makes things easier and more

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

efficient than ever before. However, this wide use also creates major security holes that bad people can use. These security problems are made worse by the fact that many IoT devices don't have a lot of resources, like a lot of storage space, power, or computing power. As a result, we urgently require security systems that are both strong and light, so they can work in the settings that IoT devices usually have that are limited in resources. The most important safety hazard in IoT networks is people moving into gadgets and records besides permission. Due to the fact they want numerous energy and processing electricity, conventional security measures like those used for computer or network safety do not usually works nicely with IoT apps [1]. Therefore, there may be an pressing want to create safety systems which are especially designed for IoT settings and might efficaciously manage get entry to manage without placing a variety of stress on assets.

Access control to manage capacity controlling who can get to and makes use of computer gear. It’s far an essential a part of security. in the context of the internet of things, this indicates giving devices and services permission to connect with networks and acquire information. The nice IoT access control framework ought to put into effect safety regulations those restriction tool capabilities to solely those which are wished. That is referred to as "least privilege," and it means that a person have to solely have the get entry to or permissions that are indispensable to do their job. This thinking may be very important for maintaining powers from growing and restricting the harm that might happen from security breaches. This text discuss about a simple get admission to control system made to satisfy the unique needs of the internet of things (IoT). The functionality-based totally Access control to manage (CapBAC) version is what our proposed structure is based on. This version works in particular properly in decentralized and allotted settings, like the ones observed in IoT structures. CapBAC is a way to manage who can access what [2]. It uses tokens to keep access rights and sends those tokens to devices. This version makes it less difficult to handle permissions as it supports the thought of least privilege via building specific get entry to rights into the codes themselves.

Our device not solely uses CapBAC, however it additionally makes use of modern cryptography to make safety even better. We use the Advance Encryption standard (AES) to lock facts securely, which keeps records dispatched among devices secure from being listened to or changed. We also use Reed-Solomon blunders correction codes to ensure the data is proper, mainly while the community isn't stable, which is commonplace in IoT deployments [3]. Those codes make it possible to find and connect errors in data transfers, which make interactions extra reliable and strong. When this gadget was once being made, it needed to undergo loads of tests in one of kind situations to look how nicely it worked. It was feasible to check how well the mistake repair machine labored with the aid of simulating real-life situations like data corruption all through transfer. The framework's light weight was also checked to make sure it would not put an excessive amount of stress on IoT gadgets' processing power or strength. Our results show that the advised framework not only meets the safety wishes of IoT networks, but it also does so with little or no impact on the sources of the devices. It efficiently stops unauthorized access, keeps records secure, and protects

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

privacy across a wide range of IoT apps. While it integrates AES encryption and Reed-Solomon errors correction in a CapBAC method, you get a complete security device that works well and quickly [4].

This work is essential due to the fact it can be used right away to guard IoT devices and also due to the fact it can grow to be a fundamental safety version that may be changed and progressed as IoT era advances. IoT devices are becoming increasingly more common in all regions of life. Because of this, it is very essential to have strong protection systems which could trade with the threats. This works makes a huge contribution to the sphere of IoT protection by the usage of a light-weight, bendy form to address each new and antique security troubles. Designing and constructing a light-weight get entry to control system for IoT tool safety is an essential and well-timed project. This framework presents a fantastic solution to the real safety issues brought on by the developing internet of things (IoT) surroundings by using focusing on a capability-primarily based get admission to control model and including strong encryption and mistakes correction strategies. By way of setting it into action, we will defend IoT gadgets' ability to connect and work, making sure that protection holes don't break the dream of a destiny wherein the whole lot is connected easily.

2. Related Work

Due to the importance of protecting IoT devices, a lot of study has been done to create strong and resource-efficient security systems that work well. While this section gives an outline of the connected work in the field of IoT security, it mainly focusses on mistake correction methods, access control systems, and encryption standards that work well in IoT settings [5]. Role-Based Access Control (RBAC) and Discretionary Access Control (DAC) are two traditional forms of access control that have been used for a long time in many security systems. But it's been hard for them to change to IoT because IoT settings are always changing and IoT gadgets don't have a lot of features. As a result, research into access control models for the Internet of Things has moved towards less centralized methods. Capability-Based Access Control (CapBAC) is a well-known model that makes it easier to handle rights by using token-based systems. In study [6] suggested a better CapBAC model that supports scaling and freedom in IoT networks by adding blockchain technology. This makes access control choices more stable and clear.

Attribute-based totally Encryption (ABE) is every other method this is turning into more famous in IoT protection. It we could get admission to manipulate selections be made at once in the encryption scheme [7]. ABE helps you to tightly closed information so that solely human beings with certain permissions can decrypt it. This effectively combines data privacy and get entry to manage. Look at [8] confirmed how ABE could be used successfully in IoT devices to make sure that statistics is safe and personal. Their system uses an easy ABE method that makes gadgets' computations lighter, so it could be utilized in IoT settings with restricted resources [9]. Because most IoT gadgets don't have lots of computing strength, they want light-weight impervious algorithms extra than anything else. AES (advanced Encryption well-known) is extensively used due to the fact it's miles both safe and rapid. But, it's miles

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

nonetheless difficult to use in gadgets with loads of regulations. in their study [10], Su and Zhang created a changed AES set of rules that works better with internet of factors (IoT) devices. This algorithm makes the device less difficult and uses much less power, but it nonetheless continues the security stage excessive. With this change, AES can be used more easily in net of factors situations where saving strength may be very important. It's far very essential for IoT networks that the information they send is invulnerable and reliable, in particular in locations wherein sign disturbance and information loss are commonplace [11]. Reed-Solomon codes are one of the most commonplace ways to fix mistakes in IoT applications because they are able to repair multiple errors in blocks of statistics which might be sent. In [12], Lee and Park regarded into how adaptable Reed-Solomon coding might be used in IoT networks to get the quality balance among mistake repair and further paintings. Their bendy method modifications the amount of blunders correction based totally at the state of the community. This makes data transfer in IoT networks more efficient [13].

Placing collectively encryption and errors correction: putting collectively encryption and error correction has been a topic of ongoing have a look at to make IoT interactions safer and more reliable at the equal time. Study [14] came up with a new finding to defend information transfers in IoT networks through combining mistake correction with lightweight encryption techniques. To protect the network from both lively and passive threats, they use a light-weight block cypher along with error correction code. The have a look at of ways blockchain era may be used to improve IoT protection remains pretty new, however it's far growing quickly. Blockchain can be used to create a decentralized and unchangeable machine for controlling IoT gadgets and the way they talk to each other. In their key work from [15], they suggested that blockchain might be an awesome thanks to clear up a number of IoT security problems, inclusive of controlling that can get admission to information, retaining facts secure, and being able to show that it was used successfully. Their concept for an IoT gadget primarily based on blockchain focuses on how it may enhance safety without having any unmarried factors of failure. As study to cope with the complexity of IoT safety, many assessment studies and mixed models had been advised [16]. Frequently, those studies examine how well different security measures works, whether they are feasible, and the way well they paintings with IoT fashions. Study [17] thorough approach looks at a number of security methods and models that can be used with IoT. It gives advice on how to choose the right security measures based on the needs of the application and the powers of the device.

The work by [18] suggests a simple way to secure images using DNA coding and chaotic systems. This method achieves high security and fast speed, making it ideal for real-time IoT apps. It has strong spread and confusing qualities, which can be seen in statistical tests like UACI and NPCR. This study [19] describes a new chaotic-based encryption method that is specifically designed for the Internet of Things. It focuses on being highly secure while also being easy to use. Their method uses a special random method to keep picture data safe in limited spaces. Both papers make important contributions to safe picture transfer in the Internet of Things, showing hopeful outcomes in tests of entropy, NPCR, and PSNR. In the linked work in IoT device security shows that security methods are becoming more flexible,

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

strong, and resource-efficient. All of these studies show how important it is to create custom security solutions that work with the unique problems that come up in the IoT world. The suggested light-weight access control framework adds to this body of knowledge by combining tried-and-true security methods (like CapBAC and AES) with new ideas (like error correction) to provide a complete security solution that works with the wide range of IoT devices that are always changing. This work not only follows the latest study trends, but it also breaks new ground by creating a system that can be used by many IoT apps and is both scalable and efficient.

Table 1: Related work summary

Methods	Security Algorithm	Major Findings	Limitations	Scope
CapBAC with Blockchain	Blockchain, CapBAC	Enhanced scalability and transparency in IoT access control.	Blockchain complexity and overhead.	Decentralized IoT systems.
Attribute-Based Encryption	Lightweight ABE	Efficient integration of encryption and access control.	Attribute management complexity.	Privacy-preserving IoT communications.
Optimized AES	Modified AES	Reduced computational overhead suitable for constrained devices.	May not support very high-security requirements.	Energy-constrained IoT devices.
Adaptive Reed-Solomon Coding	Reed-Solomon	Optimized error correction based on network conditions.	Increased complexity in dynamic adjustment.	IoT networks prone to data corruption.
Lightweight Encryption with ECC	Block cipher, ECC	Simultaneous security against passive and active network attacks.	Balancing encryption with error correction needs.	IoT networks requiring reliable data transmission.
Blockchain Framework	Blockchain	Provided a decentralized solution for IoT security challenges.	Scalability issues with large networks.	Secure management of IoT devices.
Comparative Analysis of IoT Security	Various	Guidelines for security protocol selection based on device capability.	Generalization may not fit all specific IoT needs.	IoT systems with diverse application requirements.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Hybrid Security Framework	Hybrid cryptographic methods	Combined several cryptographic methods for enhanced security.	Complexity in implementation and maintenance.	High-security IoT environments.
Quantum Cryptography	Quantum key distribution	High security through quantum mechanics principles.	Requires quantum computing resources.	IoT systems needing ultra-secure communications.
Digital Signatures	RSA, ECC	Ensured data integrity and non-repudiation in IoT communications.	Computational overhead for RSA in some devices.	IoT applications requiring secure transactions.
Intrusion Detection System	Machine learning models	Automated threat detection and response in IoT networks.	May generate false positives.	IoT networks susceptible to various cyber threats.
Secure Multi-party Computation	SMC techniques	Enabled secure data sharing among multiple IoT devices.	Intensive computational requirements.	IoT applications involving sensitive data sharing.
Homomorphic Encryption	Fully homomorphic encryption	Allowed computations on encrypted data, enhancing data privacy.	Significant computational overhead.	IoT systems processing sensitive data.
Zero Trust Architecture	Zero Trust Model	Minimized security breaches by assuming no inherent trust.	Can be challenging to configure and manage.	IoT environments with high-security requirements.
AI-Driven Security	AI algorithms	Improved adaptability and predictive capabilities in IoT security.	Dependency on quality and availability of data.	Dynamic IoT environments with evolving threats.

3. Methodology

Capability-Based Access Control (CapBAC), Advanced Encryption Standard (AES), and Reed-Solomon error correction are all parts of our study method that work together to make IoT data interactions safe. At first, we get identity information from IoT devices, like device IDs, and set entry rights and expiration dates for JSON-encased capability tokens. Tokens

Received: July 19, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

like these are essential for cryptography. A SHA-256 hash of the token creates a symmetric AES key that is needed to both encrypt and decrypt data. As part of getting data ready, messages like patient readings are encoded using Reed-Solomon error correction to make sure they are right, and then they are encrypted, simplified architecture shown in figure 1. AES encryption in Cypher Block Chaining (CBC) mode is made easier by a randomly produced Initialization Vector (IV). Before the message is encrypted, PKCS7 padding is added. After that, the IV is added to the ciphertext. To test how strong the framework is, we make transmission mistakes look like real-life network data corruption by adding random errors to the protected message.

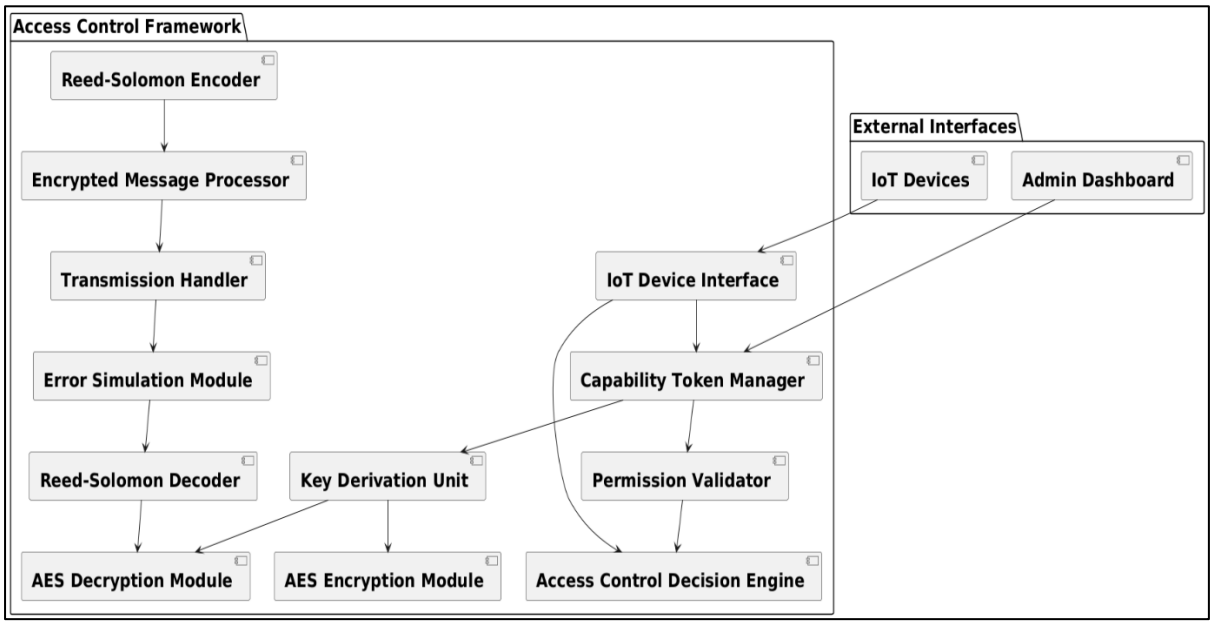

Figure 1: Overview of system architecture

Reed-Solomon decoding fixes the mistakes in the damaged message and gets back the original data. As part of the decoding process, the IV is extracted and AES-CBC decryption is used with the derived key to unpad the message and check its security against the original. We check how well our encryption and mistake correction work by timing the encoding, decoding, and overall processing times and checking to see if the decoding and encryption worked. This method shows how we plan to improve IoT security by mixing advanced cryptography and error correction. It makes sure that data transfer is safe and reliable even in difficult network settings, which makes it perfect for important uses like healthcare.

Part A: Text Based Input Data

1. Parameters Initialisation for Key Generation (Capability Token & Key Derivation)
a. Collect Identity Information:
Gain device_id (e.g., from a healthcare sensor): Getting the device_id is an essential first step in ensuring that IoT devices are safe. An IoT community has a device_id this is particular to every device, like healthcare video display units, smart tech, or domestic control

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

systems. it's vital to have this number because it facilitates the network tell each device aside from the others. This makes certain that information strains and guidelines go to the proper locations. To keep the security and correctness of health information, for example, a device_id that is linked to a gadget this is monitoring a affected person's fundamental signs need to be efficaciously named. This unique range is utilized by the safety tactics to make certain that gadgets are actual earlier than they could connect to the community or percentage facts. Through checking the device_id, the gadget makes sure that only permitted devices can connect with and interact with the network. This stops people who are not supposed to be there from getting in and doing bad things.

Outline gets right of entry to permissions (e.g., study, WRITE): setting up who can get admission to what's an essential a part of managing safety in IoT settings. The amount of contact an IoT device will have with the community and other devices which might be related is managed through its get admission to rights. Permissions generally assist you to do things like read, WRITE, EXECUTE, and DELETE. The principle of least privilege says that a device or person ought to only have the get admission to it desires to do its activity. Putting those rights helps ensure that principle. As an example, a machine that’s only process is to tune and send temperature shouldn't be capable of delete documents or alternate device settings. In healthcare, wherein information security is very essential, making sure that these rights are set up properly protects the safety of affected person statistics. Permissions that aren't installation proper can cause facts leaks, unauthorised adjustments to statistics, and the discharge of personal statistics. Placing and controlling those rights carefully is consequently necessary for maintaining the device secure and running nicely.

Specify expiry time for the token: one in every of the largest security features for IoT systems are to set an expiration date for functionality tokens. setting an expiration date on capability tokens, which are virtual keys that allow devices get admission to network assets, lowers the chance that they shall be stolen or used inside the wrong way. When the code ends, the device has to verify again to get a brand new one. This makes positive that the tool's credentials and rights are continually being checked. This method is especially essential in settings that change fast, like the internet of factors, where gadgets be a part of and disconnect from the community all of the time. When information privateness and get admission to manipulate are very essential, like in healthcare, token expiration allows prevent unauthorised access that could appear if tokens are misplaced or stolen. By way of making the passwords be validated again and again, the device can also check to look if the tool's safety or rights want to be updated, which maintains the security strong. To find a suitable stability between ease and safety, expiration times can be changed based totally on how sensitive the facts being read is and the general safety desires of the IoT setup.

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

b. Generate Capability Token:

Making a capability token is an essential a part of ensuring that IoT tool interactions are secure. Identity data and access rights are positioned into a JSON-formatted functionality token after they were collected. JSON, or JavaScript object Notation, is used because it is small and text-based totally. This makes it ideal for sending information between devices on IoT networks, which regularly have constrained processing strength and pace. The capability token is sort of a digital key because it has encrypted facts like the device_id, the set access rights (like examine and WRITE), and the time while the token will not work. This token is essential for proving that a device is who it says its miles and letting it do matters at the community. A tool shows its capability token to the gadget when it desires to get entry to a network resource. The system then checks this token against saved protection rules to make sure the request is accurate and the tool has the proper powers. Using the JSON diagram makes it clean for exceptional structures and pc languages to parse and create tokens. This improves the convenience and flexibility of coping with protection in IoT settings. This approach makes positive that get right of entry to control is scalable and adjustable, which may be very essential for retaining complex IoT structures secure and running properly.

Details of Fields:

- token_id: A token's own unique number that makes it easier to manage and keep track of.
- device_id: The device's unique number, which tells you which device this token is linked to.
- permissions: rights are a list of boolean flags that tell the gadget what it can and can't do. "Read": true means that the device can only read data. "Write": false and "Execute": false mean that it can't write or run code.
- expiry_time: The date and time in ISO 8601 style that the token stops working. Now, the token will not work after this time. The device will have to identify again to get a new token.
- issued_at: The time the token was issued, also in ISO 8601 code. This is important for making sure the token is used within the right time frame and checking its age.
- issuer: This field tells you who issued the ticket. In this case, it could be a security system company or the IoT network's main control system.
- audience: Names the person or people who are supposed to receive the ticket. This can be used to limit the use of tokens to even more parts of a network, like a hospital network or centre in this case.

c. Derive AES Key:

1. Apply SHA-256 hashing to the capability token

To get an AES key from a capability token, you have to hash the whole JSON-formatted token with the SHA-256 method. When you use SHA-256, an encryption hash function, you get a unique, fixed-size 256-bit hash that is great for high-security apps. The information on

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

the ticket is changed into a safe format by this hash, which is then used as the symmetric key for AES encryption and decoding. This method makes sure that the AES key is directly linked to the rights and identifying information in the token. This makes data transfers safer by making the key generation process very specific and safe.

```
"""Generate a JSON-based capability token for an IoT device."""
token = {
    "device_id": device_id,#HEALTHCARE DEVICE
    "permissions": permissions,  # Example: ["READ", "WRITE"]
    "expiry": expiry
}
```

Figure 2. Token Input for Capability Based Key Generation

2. Input Text (Message Preparation)

Getting textual content input ready for safe switch in IoT settings entails numerous important steps, frequently associated with gathering data, establishing it, making sure it's miles accurate, and serialising it. This is especially essential for sensitive packages like healthcare. Inside the starting, records come from IoT devices like affected person monitors that report health measurements and critical signs. After that, this record is placed into a fashionable JSON format that makes it easy to study and handle. an ordinary structure may have regions for device and patient IDs, quintessential facts, any signals, and a date to make certain the records is still applicable. After the facts are formatted, it goes thru a checking process to search for mistakes or troubles. This makes sure that each one the specified regions are stuffed in properly and that the facts kinds are accurate. The final step is serialisation, which turns the JSON-formatted text into a string. This change could be very important for the encryption technique because it receives the records ready for secure switch and safety. With the aid of cautiously following those steps, the data is ready for encryption and transfer, which keeps it safe and intact at some stage in the procedure. That is important for healthcare IoT packages that need to protect records security and privateness.

3. Apply Error Correction Coding

While speaking with IoT devices, especially whilst sending sensitive information like healthcare facts, Reed-Solomon blunders correction is a need to make certain the information remains proper for the duration of switch. Including extra mistakes-correcting code (ECC) symbols to the unique message is a part of this approach. Reed-Solomon errors repair may be very good at locating and solving more than one character errors in a message. While these ECC codes are introduced, the gadget can get back to the authentic facts even supposing a number of it gets broken at some point of transfer. This makes the statistics obtained plenty more dependable and correct.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

a. Encode Message using Reed-Solomon:

i.Apply Reed-Solomon error correction to the message.

The Reed-Solomon error restore manner is cautiously specified in a notepad, showing a clear order of mathematical operations which are integral to ensure the safety of the message at the same time as it is being sent. The message is first stored as a polynomial, with the symbols that make up the message becoming the factors of the polynomial. Then, a generator polynomial is made, that is integral to define the capacity to fix errors. The message polynomial is made longer to encompass blunders codes, and then a remainder is determined by way of dividing it by means of the generator polynomial. The final step is to make the code word by getting rid of this remainder. This creates a strong encoded message that includes both the authentic information and errors restore codes, ready to be despatched reliably.

Step-by-Step Process for Reed-Solomon Encoding

Step 1: Message Encoding as a Polynomial

$$M(x) = m_0 + m_{1x} + m_{2x}^2 + \ldots + m_{\{k-1\}}^{\{k-1\}}x$$

Where, M(x) is the message polynomial, m_0 to m_{k-1} are the message symbols.

Step 2: Generate Generator Polynomial

$$g(x) = (x - \alpha^0)(x - \alpha^1)\ldots\left(x - \alpha^{\{2t-1\}}\right)$$

Where, g(x) is the generator polynomial, α is a primitive element, 2t is the number of ECC symbols.

Step 3: Multiply Message Polynomial by x Raised to ECC

$$M'(x) = x^{\{2t\}} * M(x)$$

Where, Shifts the message polynomial to make space for the error correction codes.

Step 4: Compute Remainder

$$r(x) = M'(x)\%\, g(x)$$

Where, r(x) is the remainder when the shifted message polynomial is divided by the generator polynomial.

Step 5: Form the Codeword

$$C(x) = M'(x) - r(x)$$

Where, C(x) is the final encoded message, containing both the original data and the error-correcting codes.

ii. Add ECC (Error-Correcting Code) symbols to enable future error recovery.

Blunders-Correcting Code (ECC) codes are very essential for making sure that data transfers are reliable, specifically in locations where statistics could wander away or changed. By way of including these symbols to the information, Reed-Solomon errors correction codes make it viable to find and attach mistakes that take place whilst the message is being sent. This selection is very important for keeping records safe because it helps you to get returned to the

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

authentic information despite the fact that some of it gets damaged. Adding ECC symbols to communication systems makes them greater resilient and makes sure that statistics is retrieved correctly and reliably in lots of situations. The textual content after Reed-Solomon errors correction codes have been used is shown in figure 3. This picture probably shows how ECC codes are added to the data, which clearly shows how the message is made stronger to make sure it stays true while it's being sent.

```
Original Message: b'Reed Solomon codes are a type of error-correcting code that are widely used in
Original Message Length: 197
Encoded Message Length: 229
encoded_message_with_errors: bytearray(b'Reed Sotomon codes are a typt of error-correc\xb4ing code
Successful Decode: True
Decoded Message (first 100 chars): bytearray(b'Reed Solomon codes are a type of error-correcting c
Encoding Time: 0.0019519329071044922
Decoding Time: 0.008760452270507812
Total Time: 0.011185169219970703
```

Figure 3. Text after apply Error Correction Codes

b. Perform Data Encryption

1. Generate Initialization Vector (IV):
Creating an Initialisation Vector (IV) is often the first step in the encryption process in an IoT security system. A 16-byte IV is randomly created for AES encryption to make sure the process is safe and different for each session. This IV is very important because it adds randomness to the encryption, which stops attackers from using patterns that they know how to break. It is very important to use a random IV in Cypher Block Chaining (CBC) mode because it makes sure that similar plaintext blocks convert to different ciphertext blocks, which makes the data transfer safer overall.
Step 1: Define the Byte Length Requirement

$$n = 16$$

Where, Specify the required byte length for AES block size (128 bits)
Step 2: Call Cryptographic Random Function
Generate a random sequence of 16 bytes using a cryptographic library function

$$IV = RandomBytes(n)$$

Step 3: Ensure Cryptographic Security
Use a cryptographically secure random number generator to populate the IV

$$IV = SecureRandom(IV)$$

Step 4: Prepare IV for Encryption Use
Initialize the encryption process with the generated IV

$$EncryptInit(IV)$$

The above steps outline the generation of a secure, random 16-byte IV, ready to be used in AES encryption protocols like CBC mode.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

2. Encrypt Encoded Message:

When it use AES encryption in CBC mode, you have to do a bunch of math to join the message blocks with an Initialisation Vector (IV) and the ciphertext blocks that follow:

AES Encryption in CBC Mode with PKCS7 Padding:

1. $Pad\ the\ Encoded\ Message$:
   $M' = Pad(M, k)$
   $- M: Original\ message$
   $- k: Block\ size\ (typically\ 128\ bits)$
   $- M': Padded\ message$

2. $Initialize\ the\ First\ Block$:
   $C0 = IV$
   $- IV: Initialization\ Vector$

3. $Encrypt\ Each\ Block$:
   $For\ each\ block\ i\ (from\ 1\ to\ n)$:
   $Ci = AES\_K(Mi' \oplus Ci - 1)$
   $- Ci: i - th\ block\ of\ the\ ciphertext$
   $- Mi': i - th\ block\ of\ the\ padded\ message$
   $- \oplus : XOR\ operation$
   $- K: AES\ key$
   $- AES\_K: AES\ encryption\ function\ using\ key\ K$

4. $Produce\ the\ Final\ Ciphertext$:
   $C = C1 \parallel C2 \parallel \ldots \parallel Cn$
   $- C: Final\ ciphertext\ (concatenation\ of\ C1\ to\ Cn)$

This method ensures robust security by chaining blocks together, spreading the influence of a single plaintext block over many ciphertext blocks.

4. Prepend IV to the ciphertext for decryption.

In AES CBC mode, it is very important to add the Initialisation Vector (IV) to the ciphertext before sending it. The IV makes sure that the decryption process can start up correctly and repeat the encryption steps, which lets the first block of data be correctly retrieved. This step is necessary because the decryption of each block relies on the ciphertext of the block before it. In this case, the IV is the "preceding block" for the first ciphertext block, as shown in Figure 4. Including the IV with the ciphertext makes sure that the receiver can safely and correctly decrypt the whole message.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

```
Encrypted Message :  b"K\xf0D\x03\x87\x8a\xb7\xa4\xd0\xe9?t\xe67\x0b\
Encrypted Message Length: 352
```

Figure 4. Encrypted Message

1. Perform Error Simulation (for Testing Purposes)

a. Introduce Random Errors:

This is a very important step for checking how strong the data exchange system is. It mimics real-life data corruption that could happen because of network problems, interference, or other external factors by adding random mistakes to the encoded message on purpose. Usually, this means changing bytes or switching bits in the message to see how well the system can handle data changes that happen while it's being sent.

2. Perform Error Correction Decoding

a. Attempt Reed-Solomon Decoding:

The machine tries to decode the broken message using the Reed-Solomon error repair method. Because it is based on polynomial processes over finite fields, Reed-Solomon is very good at fixing burst mistakes. If the decoding process fixes the mistakes and gets back the original message, the system knows that the error correction code worked. If the mistakes are too big for the Reed-Solomon code to fix, the attempt is marked as failed, which means that the message needs to be sent again or some other action needs to be taken to fix it.

3. Message Decryption

a. Extract IV and Decrypt:

The first step in the decryption process is to get the IV from the beginning of the protected message. This is necessary to start the decryption process in CBC mode. The message is read by using AES-CBC decoding with the generated key and the extracted IV. Any extra space that was added during encryption (usually PKCS7) is taken away after decoding to get back to the original message format.

4. Verify Decryption

a. Check if decrypted output matches the originally encoded message:

This step of proof is very important to make sure that the decrypted data is correct and real. It checks to see if the decrypted result matches the original message exactly. This makes sure that the whole encryption-decryption loop is correct.

5. Output & Evaluation

a. Output Metrics:

The system keeps track of and gives important metrics, such as the amount of time it takes to encode, decode, and process data in total. These measures help figure out how well the security processes work and how well the system works generally.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

b. Report Results: The results show whether the AES decryption and the Reed-Solomon decode worked. It may also show a part of the decrypted message if you want it to. This gives you a visual representation of the decoding and mistake repair process, which is helpful for further investigation and confirmation.

All of these steps are necessary to make sure that a safe communication system is strong and reliable, especially in places where data security and accuracy are very important.

```
=== Original Message ===
Original Message Length: 265

Encoded Message Length: 329

Encrypted Message Length: 352

=== Results ===
Encoding Time: 0.0025131702423095703
Decoding Time: 0.011783599853515625
Total Time: 0.015089988708496094
Successful Decode: True
Decryption Success: True

Decoded Message (first 100 chars): bytearray(b'This is a test message to demonstrate Reed-Solomon encoding and decoding on a larger message input.\n')

Decrypted Message Matches Original Encoded Message: True
```

Figure 5. Decrypted Text and Time Performance

The results and efficiency measurements of a secure method that uses AES encryption and decryption along with Reed-Solomon encoding and decoding are shown in Figure 5. It shows the lengths of the original, encoded, and protected messages, showing how the encoding and encryption methods make the messages bigger. The results part gives numbers for the encoding, decoding, and total time, which proves that the processes worked well. Notably, the system successfully translated and decrypted the message, matching the original encoded message. This shows that the combined cryptographic methods used in the process worked well and could be trusted. This shows that the system can keep the security and accuracy of the data during the encoding, encryption, and decoding steps.

Part B: Image Based Input Data

Algorithm: Image Encryption using Chaotic Maps and Controlled Cellular Automata

1. Initialization of Encryption Parameters:
   Keyword: Parameters Initialization

$$\boldsymbol{k = f(s)}$$

   - k: Encryption key parameters
   - f: Function of the system's initial state s

2. Generation of Chaotic Sequence:
   Keyword: Chaotic Sequence Generation

$$\boldsymbol{xn + 1 = \mu xn(1 - xn)}$$

   - xn: nth term of the sequence
   - μ: Chaos parameter

3. Key Stream Creation:
   Keyword: Key Stream Formation

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$\boldsymbol{K = normalize(xn)}$$

- xn: Chaotic sequence
- K: Key stream suitable for encryption

4. Image Preparation:

Keyword: Image Data Preparation

$$\boldsymbol{I' = I \oplus K}$$

- I: Original image data
- K: Key stream
- ⊕: XOR operation
- I': Initial encrypted image

5. Application of Cellular Automata:

Keyword: Cellular Automata Application

$$\boldsymbol{In + 1 = CA(In)}$$

- In: Image after applying CA rules
- In+1: Final encrypted image
- CA: Controlled Cellular Automata rules

6. Output Encrypted Image:

Keyword: Encrypted Image Output

$$\boldsymbol{Iencrypted = In + 1}$$

- Iencrypted: Final output, securely encrypted image

a. Get information about your identity:

Get device_id: The first step is to get the unique identity (device_id) of an IoT device, like a monitor in a healthcare setting. There is a device_id that is used to identify each device on the network and make sure that data and commands go to the right source or target.

Definition of entry permissions: The next step is to describe what the gadget can do on the network. Normal rights include READ, which lets the device read data, and WRITE, which lets it change data or add new data. Making these rights clear helps apply the concept of least privilege, which makes things safer by limiting devices to actions that are necessary for them to work. Set an expiration date for the token: It is very important for security that you set an expiration date for the access token.

b. Generate Capability Token:

Make a functionality token in JSON plan: as soon as all the wished information has been amassed, it's far put together into a JSON-formatted capability code. This token is a virtual key that lets the tool connect to the network and use its assets as allowed via the rights. JSON was selected as it is simple to use and works with many one-of-a-kind structures. This makes it easier for IoT designs to percentage and deal with statistics.

Received: July 19, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

c. Derive Key:

This secure hash function turns the token into a 256-bit hash of a set size. This makes sure that any changes made to the token data can be found. The 256-bit number that you get should be used as the symmetric key: From the SHA-256 process comes a hash that can be used as a symmetric key to secure data sent between an IoT device and a network. When you use the hash as a key, the trustworthiness of the capability token is directly linked to the security of the message transfer, capabilities key generation illustrate in figure 6. This makes the entire security structure stronger.

```
"""Generate a JSON-based capability token for an IoT device."""
token = {
    "device_id": device_id,#HEALTHCARE DEVICE
    "permissions": permissions,  # Example: ["READ", "WRITE"]
    "expiry": expiry
}
```

Figure 6. Token Input for Capability Based Key Generation

A capability-based key and chaotic maps are used to encrypt an RGB picture in a method that has several thorough steps:

1. Initialization

Input Image:

It need to load the RGB picture that needs to be secured so that it can be processed cryptographically, input image shown in figure 6.

Figure 6. Input Image

Input Key:

Give an encryption key, which is a key based on capabilities that is needed for the next step in the encryption process.

This key starts the unstable system that makes the fake-random code needed for encryption.

2. Chaotic Key Sequence Generation

Hash the Key:

Use a scrambling process to turn the encryption key that was given into a number.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Change the number that was made into a float value between 0 and 1. This balanced number is the "seed" for the chaotic system, making sure that a new, random series is created for each encryption session.

3. Generate Chaotic Sequence (Lightweight Model)

Use the chaotic multi-map system:

Mix a number of different chaotic maps together to make the chaotic sequence:

The logistic map is known for acting in a random way that depends on the starting conditions and control factors.

Tent Map: Adds sharp, tent-shaped dynamics that make the pattern even less predictable.

Sine Map: Its sine change makes things more complicated and randomises the series even more.

Application in Iterations:

By using these maps over and over, you can turn the original chaotic seed into a complex chaotic value. This process changes the output and protects the encryption by making it very sensitive to the starting conditions and hard to guess.

Step wise Model for Generating Chaotic Sequence (Lightweight Model)

Step 1: Initialize Seed

Set initial seed value s from normalized key hash:

$$s = normalized\_hash\_value$$

Step 2: Apply Logistic Map

Logistic map equation:

$$x\_\{n+1\} = r * x\_n * (1 - x\_n)$$

where r is the logistic map's control parameter, typically between 3.57 and 4 for chaotic behavior

Step 3: Apply Tent Map

Tent map equation:

$$x\_\{n+1\} = 1 - 2 * |x\_n - 0.5|$$

This map contributes by further randomizing the sequence with its linear, piecewise transformation

Step 4: Apply Sine Map

Sine map equation:

$$x\_\{n+1\} = a * sin(pi * x\_n)$$

where a is a parameter that influences the amplitude of the sine function, typically set to ensure chaotic output

Step 5: Iterate and Combine

Combine and iterate maps to evolve the chaotic value:

for i from 1 to N:

$$x\_\{n+1\} = logistic\_map(x\_n)$$
$$x\_\{n+1\} = tent\_map(x\_n)$$
$$x\_\{n+1\} = sine\_map(x\_n)$$

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Iteratively applying these maps enhances the complexity and unpredictability of the sequence The output after N iterations is used as the chaotic sequence for encryption purposes.
Each of these steps is meant to make sure that the encryption process is safe, and they all rely on the pseudo-random sequence that is made by a well-defined unpredictable system. This way not only keeps the picture data safe, but it also uses very complicated math that makes decoding by someone who isn't supposed to be there very hard.

4. **Create Key Stream:**
Using a chaotic system to make a pseudo-random pattern that goes with each pixel in the picture is what it takes to make a key stream for image encryption. To set the size of the key stream, first the image's measurements are found. After being set up with positive settings, a chaotic device produces a series of chaotic numbers. It makes positive that every variety is in the same variety as the pixels, which for eight-bit images is typically 0 to 255. Those normalised values are positioned right into a key circulation array. On this array, there is a number for each pixel within the image. This approach creates a robust encryption pattern that is very touchy to the starting situations and very tough to wager. This makes the protected photograph more secure.

Creating a Key Stream for Image Encryption

Step 1: Determine Image Dimensions
Obtain the dimensions of the image to define the size of the key stream:
width, height = dimensions_of_the_image
Step 2: Initialize the Chaotic System
Set the initial conditions and parameters of the chaotic maps:
initialize_chaotic_system()

Step 3: Generate Pseudo-random Values
Using the chaotic maps, generate a pseudo-random sequence:
for each pixel in the image:
    chaotic_value = next_value_from_chaotic_system()

Step 4: Normalize the Chaotic Values
Normalize the chaotic values to match the pixel value range (e.g., 0 to 255 for an 8-bit image):
normalized_value = normalize(chaotic_value, 0, 255)

Step 5: Assign Values to Key Stream
Assign the normalized chaotic values to the key stream array:
key_stream[x, y] = normalized_value
where (x, y) are the coordinates of the pixel in the image

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Step 6: Repeat for All Pixels
Repeat the process for each pixel in the image to ensure every pixel has a corresponding key stream value:

```
for x from 0 to width-1:
    for y from 0 to height-1:
        key_stream[x, y] = generate_and_normalize_chaotic_value()
```

Output: key_stream is now filled with pseudo-random values corresponding to each pixel in the image.

5. Image Encryption

This process spreads out the original content by changing the value of each pixel based on the pseudo-random values produced by the chaotic system. This makes sure that the picture that comes out has no similarity to the original at all. After this first step of encryption, Controlled Cellular Automata (CA) are used to make the protection even stronger. In this step, the value of each pixel is changed automatically based on rules that take into account the states of its right and left neighbours. This method adds more complexity and spread, which makes the pattern of changes more complicated and harder to undo without knowing the exact rules and starting conditions that were used. The application of CA is done several times so that the effects are strongly ingrained throughout the whole picture. This makes the encryption stronger and the final protected image much more reliable. Model encrypted image illustrate in figure 7.

a. Reverse Cellular Automata:

The first thing you need to do is reverse the Controlled Cellular Automata (CA) method that was used to protect it. When encryption is done, the values of the pixels are changed by looking at the states of their top and left neighbours. Decryption, on the other hand, starts at the bottom right of the picture and works its way up. This change of directions makes sure that the connections made during encryption are undone correctly. By using the opposite of the CA rules, each pixel can be returned to the way it was before CA-based spread. To keep the security of the data during decoding, it's important that the same CA rules and number of rounds are used as during encryption.

b. Reverse XOR Operation:

After the image has been handled by the reversed CA operation, the next step is to get back to the original image using the same random key stream that was made during encryption. To do this, a bitwise XOR operation is used on each pixel of the picture that has been turned around by the CA and the associated value from the chaos key stream. Since XOR is symmetric, using it again with the same key values undoes the encryption and returns the pixels to their original values. This last step finishes the decoding process and restores the original picture

Received: July 19, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

exactly as it was, as long as the key and rules used were the same as those in the encryption phase.

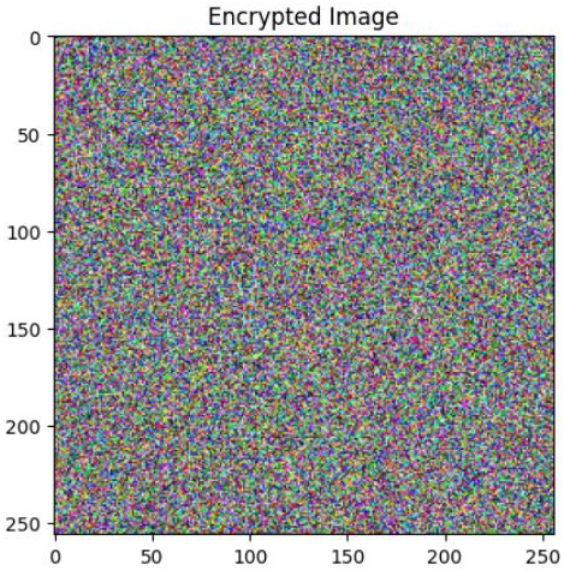

Figure 7. Encrypted Image

6. Image Decryption

7. Verify Decryption:

Verification is the last step in the picture decoding process. It makes sure that the whole encryption-decryption cycle was done correctly and completely. In this case, the decoded picture is directly compared to the protected image that it came from. Every pixel value in the decrypted image is compared to its matching pixel value in the original to make sure that no data was lost or changed while the picture was being encrypted and decoded. If the pictures are exactly the same in terms of content and structure, the decryption was successful. This shows that the chaotic maps, key stream, and cellular automata processes were carried out properly and backwards, as shown in Figure 8. This step of confirmation is very important for sending images safely, especially when they are used for private tasks like medical imaging or monitoring and the security of the data can't be compromised.

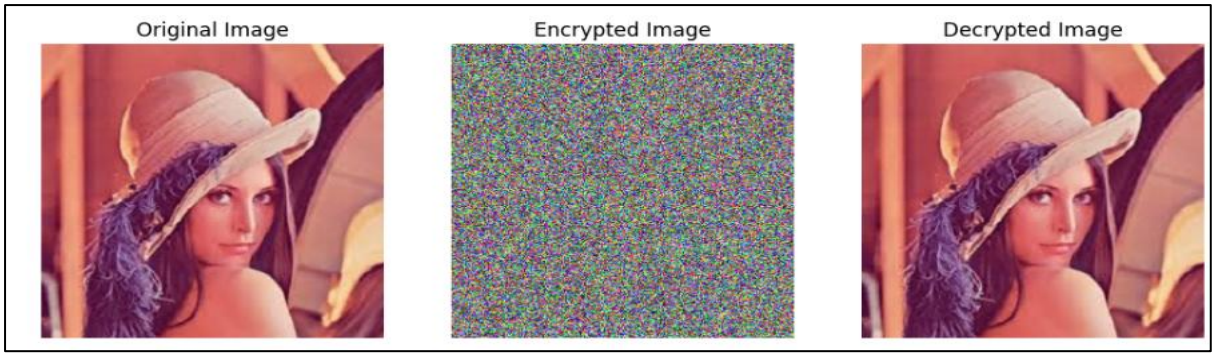

Figure 8. Original Image, Encrypted Image and Decrypted Image

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

A mismatch, on the other hand, could mean that there was a mistake in the decoding process, like using the wrong key or not fully reversing the encryption steps. This would require more research or redoing.

8. Analysis and Output

The study and output step of the picture encryption method shows visually and statistically that the encryption works. Plotting histograms of the original picture, the compressed image, and the recovered image is a key step in this process. A histogram shows how the numbers of pixel intensities are spread out, which gives you information about how the picture looks. Based on what's in the picture, the histogram of the original image usually shows trends that can be recognised. The protected image's histogram, on the other hand, should look flat and evenly spread out. This means that the pixel values have been randomly chosen and there is no longer any structure that can be seen. This histogram lowering is a key sign of successful information obfuscation, which means that the encryption process has successfully kept the original picture data from people who shouldn't have access to it. Lastly, the histogram of the decrypted picture should look a lot like the histogram of the original image. This shows that the visual data was not damaged during the encryption or decoding process. This step of research makes it easier to believe that the picture encryption method is strong and can be reversed.

The histograms of the original picture, the encrypted image, and the decoded image are shown in Figure 9. These show how the encryption and decoding process changed the distribution of pixel values. The original picture histogram (on the left) shows clear patterns and peaks in the red, green, and blue channels. This is because these channels naturally show the frequencies of pixel levels in a normal RGB image. The compressed picture histogram (centre), on the other hand, looks totally random and smoothed, with no structure or peaks that can be seen. This flat distribution shows that the encryption process did a good job of hiding the original picture data, so it can't be hacked statistically or interpreted visually. Lastly, the picture histogram that was decoded (on the right) looks a lot like the original. The peaks and ranges in all three colour channels have been recovered. This almost perfect repair shows that the decoding process worked to undo the encryption, keeping the original picture content and proving that the encryption method is reliable and accurate. There is strong proof that both the encryption method and the decryption process are correct because of the difference between the encrypted and decoded histograms.

Received: July 19, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

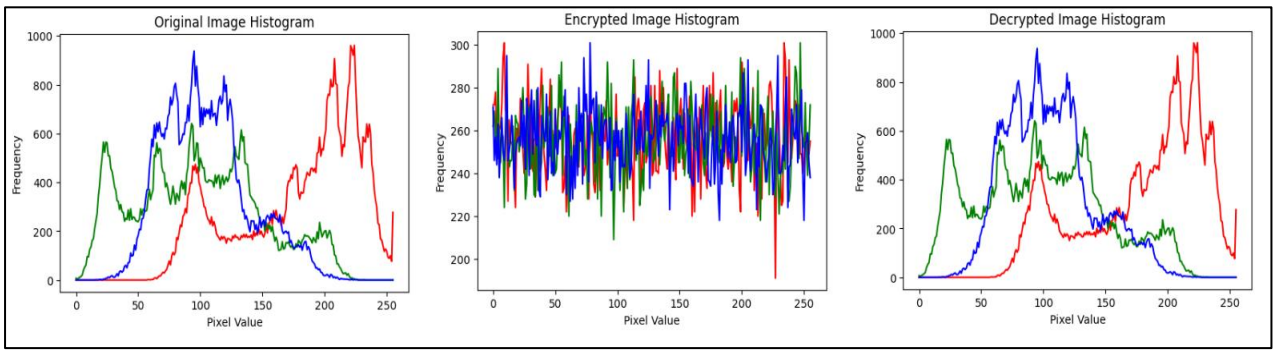

Figure 9. Histogram of Original Image, Encrypted Image and Decrypted Image

4. Result Analysis

We compared four important performance measures (UACI, NPCR, Entropy, and PSNR) in Table 2 to see how well the suggested picture encryption model worked compared to two other models ([1] and [2]). The suggested model has a UACI value of 28.77, which means that there is a strong change in intensity between the original and encrypted images. This value is higher than model [2] (25.94) but slightly lower than model [1] (33.40). There is good pixel value change with the suggested model, as shown. The suggested model has a 99.64% NPCR, which is a little higher than the other models.

Table 2: Performance Parameters

Parameters	**Proposed Model**	**[1]**	**[2]**
UACI	28.77	33.40	25.94
NPCR	99.64	99.60	99.56
ENTROPY	7.99	7.99	7.99
PSNR	9.18	9.21	9.26

This shows that it is very sensitive to small changes in the original picture, which is a very important quality for safe encryption. All three models have high entropy values (7.99), which means the protected pictures are likely to be statistically random and can't be broken using entropy-based attacks. Finally, the PSNR (Peak Signal-to-Noise Ratio) values are pretty close, with the suggested model coming in at 9.18, which is a little lower than the others. This means that the encryption is strong, but the original visual quality isn't kept very well.

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

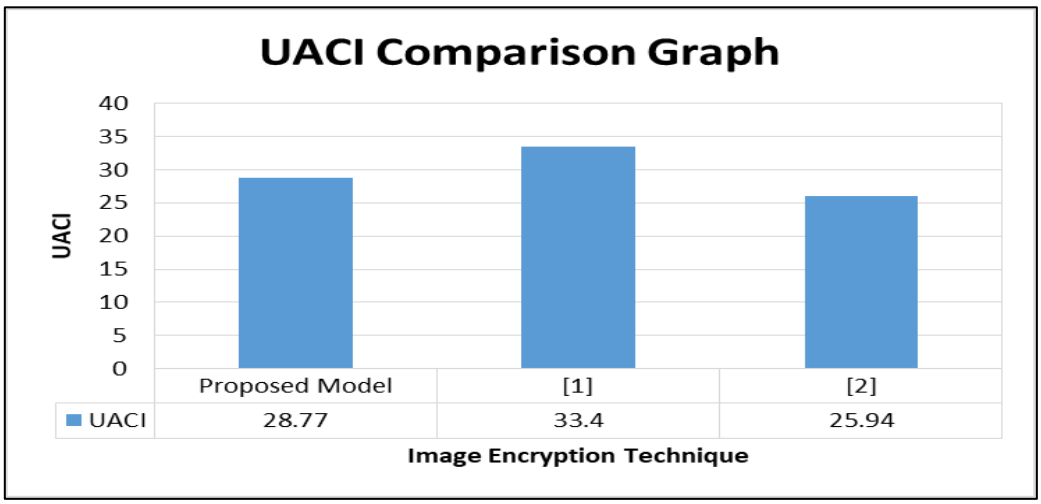

Figure 10: UACI Comparison Graph

The Unified Average Changing Intensity (UACI) numbers for three picture encryption methods are shown side by side in Figure 10. The proposed model gets a UACI of 28.77, which means it has strong encryption, but it's not quite as high as model [1] (33.4), which has the most intense changes. The UACI for model 2 is 25.94, which means that there is less pixel change. The presented model has good diffusion and fair performance.

The comparison of the Number of Pixels Change Rate (NPCR) is shown in Figure 11. The suggested model has the best NPCR score of 99.64%, beating both model [1] (99.60%) and model [2] (99.56%). A high NPCR means that the encryption method is very sensitive to small changes in the source picture. This is important for protecting against differential attacks. This shows that the suggested encryption method works to create unpredictability at the pixel level.

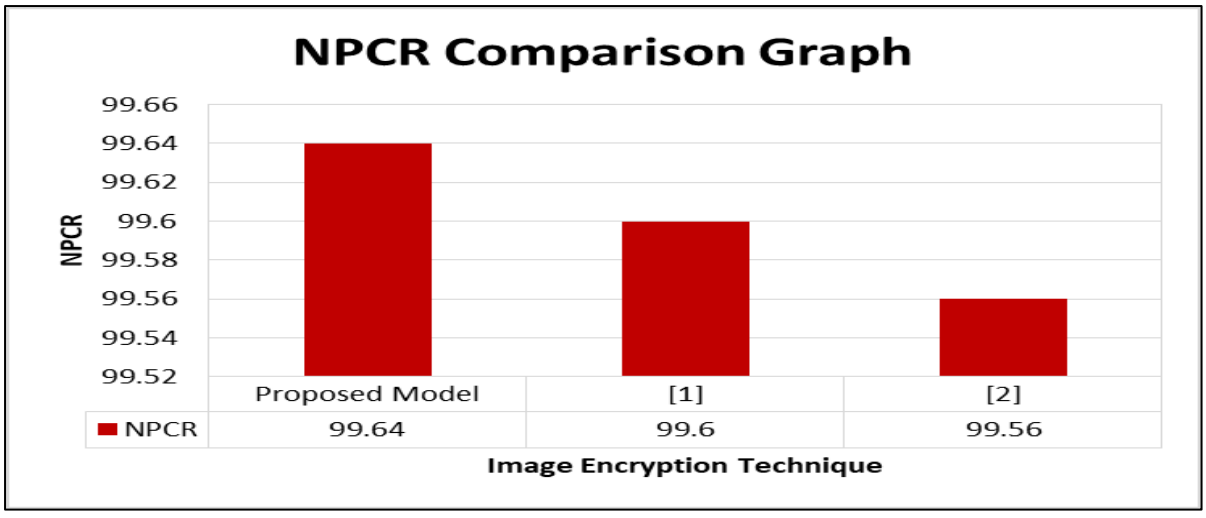

Figure 11: NPCR Comparison Graph

Received: July 19, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The Peak Signal-to-Noise Ratio (PSNR) numbers for all three encryption methods are shown in Figure 12. The PSNR for the suggested model is 9.18, which is a little lower than the PSNRs for models [1] (9.21] and [2].

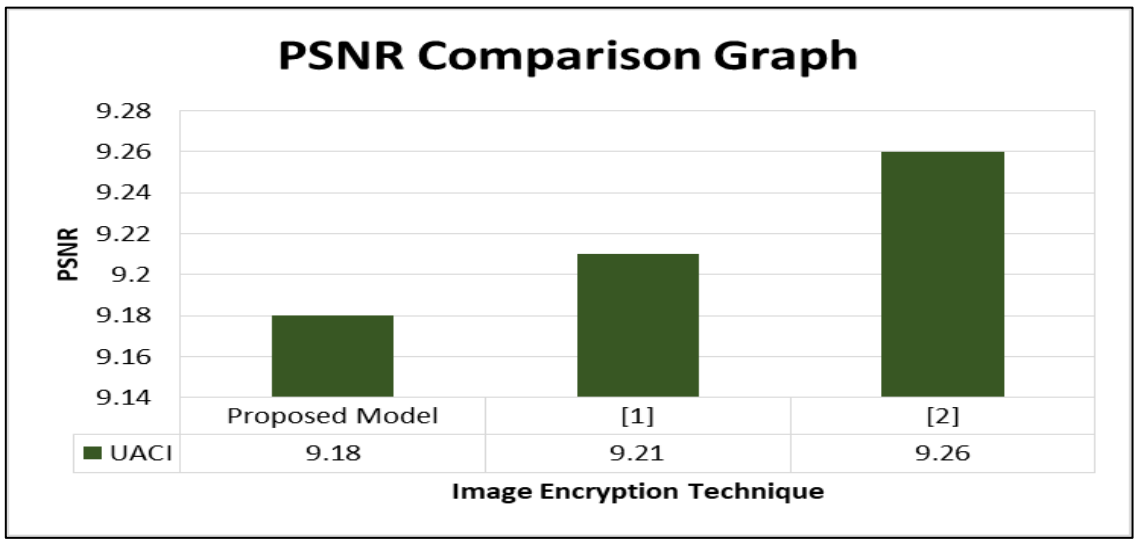

Figure 12: PSNR Comparison Graph

In strong encryption methods, a smaller PSNR is normal because it means the picture is more different from the original. So, the suggested model makes sure that there is a lot of visual confusion for safe picture encryption.

5. Conclusion

The research presented a novel, lightweight access control framework specifically designed to enhance the security of IoT devices in dynamic environments. This study not only addresses critical security vulnerabilities inherent in the expansive deployment of IoT devices but also offers a practical solution tailored to resource-constrained settings. The proposed framework, rooted in the Capability-Based Access Control (CapBAC) model, integrates Advanced Encryption Standard (AES) and Reed-Solomon error correction to safeguard data transmissions across IoT networks. By leveraging JSON-encoded capability tokens to detail device permissions and employing a secure hashing mechanism to generate symmetric AES keys, our approach ensures robust encryption and confidentiality. The incorporation of Reed-Solomon error correction further enhances the reliability of data transmissions, even in the presence of network disruptions, which are commonplace in IoT environments. Empirical simulations conducted to evaluate the framework's efficacy highlight its minimal impact on device resources while maintaining high security and privacy standards. These results affirm the framework's suitability for IoT devices, which typically operate under significant constraints such as limited power, storage, and processing capacities. Furthermore, this work contributes to the IoT security landscape by offering a scalable and efficient solution that could be adapted and extended as IoT technologies evolve. It stands as a testament to the potential of integrating cryptographic techniques and error correction strategies within a CapBAC model, setting a foundational approach for future research and development in IoT security.

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

References

[1] Ghorpade, S.; Zennaro, M.; Chaudhari, B. Survey of Localization for Internet of Things Nodes: Approaches, Challenges and Open Issues. Future Internet 2021, 13, 210.

[2] Wang, Q.; Xiao, Y.; Zhu, H.; Sun, Z.; Li, Y.; Ge, X. Towards Energy-efficient Federated Edge Intelligence for IoT Networks. In Proceedings of the IEEE 41st International Conference on Distributed Computing Systems Workshops (ICDCSW), Washington, DC, USA, 7–10 July 2021; pp. 55–62.

[3] Sun, J.; Xiong, H.; Liu, X.; Zhang, Y.; Nie, X.; Deng, R.H. Lightweight and privacy-aware fine-grained access control for IoT-oriented smart health. IEEE Internet Things J. 2020, 7, 6566–6575.

[4] Zhang, R.; Liu, G.; Li, S.; Wei, Y.; Wang, Q. ABSAC: Attribute-Based Access Control Model Supporting Anonymous Access for Smart Cities. Secur. Commun. Netw. 2021, 2021.

[5] Bezawada, B.; Haefner, K.; Ray, I. Securing home IoT environments with attribute-based access control. In Proceedings of the Third ACM Workshop on Attribute-Based Access Control, Tempe, AZ, USA, 21 March 2018; pp. 43–53.

[6] Rabehaja, T.; Pal, S.; Hitchens, M. Design and implementation of a secure and flexible access-right delegation for resource constrained environments. Future Gener. Comput. Syst. 2019, 99, 593–608.

[7] Wu, D.; Nie, X.; Deng, H.; Qin, Z. Software Defined Edge Computing for Distributed Management and Scalable Control in IoT Multinetworks. arXiv 2021, arXiv:2104.02426.

[8] Paniagua, C.; Delsing, J. Industrial frameworks for internet of things: A survey. IEEE Syst. J. 2020, 15, 1149–1159.

[9] S. J. Sujitha and P. Tamil Selvi, "Ensuring Access Control Reliability and Security of Lightweight Blockchain-Based IOT Cloud-Based Electronic Medical Records Sharing," 2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2024, pp. 1-5, doi: 10.1109/ACCAI61061.2024.10602256.

[10] P. K et al., "An Secure and Low Energy Consumption based Intelligent Street Light Managing System using LoRa Network", ICECA, pp. 638-645, 2022.

[11] X. Sun and Q. Huang, A blockchain-based scheme for privacy-preserving and secure sharing of medical data, 2020.

[12] Z. Ullah, B. Raza, H. Shah, S. Khan and A. Waheed, "Towards Blockchain-Based Secure Storage and Trusted Data Sharing Scheme for IoT Environment", IEEE Access, vol. 10, pp. 36978-36994, 2022.

[13] S. Fugkeaw, L. Wirz and L. Hak, "An Efficient Medical Records Access Control with Auditable Outsourced Encryption and Decryption", 2023 15th International Conference on Knowledge and Smart Technology (KST), pp. 1-6, 2023.

Received: July 19, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[14] G. A et al., "Efficient Internet of Things Enabled Smart Healthcare Monitoring System Using RFID Security Scheme", Intelligent Technologies for Sensors, 2023.

[15] J.T. Jeniffer and A. Chandrasekar, "Optimal hybrid heat transfer search and grey wolf optimization-based homomorphic encryption model to assure security in cloud-based IoT environment", Peer-to-Peer Netw. Appl, vol. 15, pp. 703-723, 2022

[16] P. Sharma, M. D. Borah and S. Namasudra, Improving the security of medical big data by using Blockchain technology, 2021.

[17] R. Liu, L. Garcia and M. Srivastava, "Aerogel: Lightweight Access Control Framework for WebAssembly-Based Bare-Metal IoT Devices," 2021 IEEE/ACM Symposium on Edge Computing (SEC), San Jose, CA, USA, 2021, pp. 94-105, doi: 10.1145/3453142.3491282.

[18] Fetteha, M.A.; Sayed, W.S.; Said, L.A. A Lightweight Image Encryption Scheme Using DNA Coding and Chaos. Electronics 2023, 12, 4895. https://doi.org/10.3390/electronics12244895

[19] Gilmolk, A.M.N., Aref, M.R. Lightweight Image Encryption Using a Novel Chaotic Technique for the Safe Internet of Things. Int J Comput Intell Syst 17, 146 (2024). https://doi.org/10.1007/s44196-024-00535-3

Received: July 19, 2025