Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.079018

ARTICLE

Secure IoT Data Transmission Using MPEG Derived Motion Vectors and Dual Encryption Techniques

Sara H. Elsayed[1], **Rodaina Abdelsalam**[1], **Mahmoud A. Ismail Shoman**[2], **Raed Alotaibi**[3,*] and **Omar Reyad**[4,5,*]

[1]Faculty of Computers and Information Technology, EELU, Giza, Egypt
[2]Faculty of Computers and Artificial Intelligence, Cairo University, Giza, Egypt
[3]Applied College, Shaqra University, Shaqra, Saudi Arabia
[4]College of Computing and Information Technology, Shaqra University, Shaqra, Saudi Arabia
[5]Faculty of Computers and Artificial Intelligence, Sohag University, Sohag, Egypt

*Corresponding Authors: Raed Alotaibi. Email: alhafi@su.edu.sa; Omar Reyad. Email: oreyad@su.edu.sa

Received: 13 January 2026; Accepted: 23 February 2026; Published: 08 May 2026

ABSTRACT: In today's digitally connected world, where cyber threats are becoming increasingly complex, finding modern and secure text encryption solutions that maintain maximum runtime performance while offering high-level protection is more crucial. The deployment of sophisticated security paradigms is often accompanied by a significant escalation in computational overhead. Thus, the fundamental objective resides in the mitigation of computational overhead while maintaining an uncompromising security posture. Internet of Things (IoT) devices require strong security measures for data transmission. Also, protecting communication channels against illegal access and eavesdropping has become crucial due to the exponential expansion of the IoT. The IoT implementations frequently have weak, unencrypted data streams that are susceptible to manipulation and interception. In order to overcome this, the proposed work incorporates lightweight protection using Moving Picture Experts Group (MPEG) derived motion vectors and dual encryption techniques to guarantee message confidentiality and integrity via limited IoT networks. The proposed method starts with resizing MPEG video frames to dimensions [1080, 1920]. After extracting motion vectors from two successive video frames, scale the obtained vectors to 1000. The exclusive OR (XOR) procedure is applied to the combined motion vectors. A one-dimensional (1D) vector is then produced. The initial elliptic curve Diffie-Hellman (ECDH) private key is created using a mapping of a hash function. The public keys, shared secret keys, and a second private key are also created. The shared secret key is used to generate the Advanced Encryption Standard (AES) main key. After that, the created AES is used to encrypt and decrypt text messages ranging in length from 10 to 300 bytes. Several evaluation metrics, including mean square error (MSE), peak signal to noise ratio (PSNR), correlation coefficient (CC), avalanche Effect (AE), and compression ratio (CR) values, are evaluated between the original and ciphertext. The presented method has demonstrated optimal performance in terms of encryption and decryption times as well as public and private key generation. Thus, improving the IoT application's overall security condition by guaranteeing that only authorized endpoints can decrypt and read the data, and showing minimal latency overhead as compared to insecure transmission. This suggests that it is a highly effective solution for secure text communication, offering lightweight encryption suitable for a wide range of resource-constrained and real-time applications.

KEYWORDS: MPEG; motion vectors; encryption; decryption; IoT; ECDH; AES; MES; PSNR; CC; AE; CR

1 Introduction

Nowadays, a vast proliferation of sensors and devices are connected with IoT applications to enable unprecedented levels of automation, efficiency, and data collection. Therefore, protecting IoT applications is a fundamental need rather than an optional feature. To ensure that the enormous advantages of IoT are not compromised by systemic risk, secure IoT design must incorporate strong encryption for data transfer, strict device authentication, and frequent security updates to guard against unauthorized access, data breaches, and possible physical harm. To store, analyze, process, and compute vast amounts of data, the cloud computing model is combined with the IoT. The cloud of things (CoT) is the name given to this integrated computing concept. Although it has many uses in various fields, it is unable to handle real-time and latency-sensitive applications since it could lower service quality [1]. Security is a challenging issue for IoT because of three factors: heterogeneity, resource limitations, and unpredictable environment. Heterogeneity is particularly important since a given system may contain devices from many manufacturers and possibly multiple owners, each with their own security limitations and defenses [2]. IoT applications face significant security concerns, and the majority of IoT devices are not designed to address security and privacy issues. There will be a significant rise in malicious attacks and other security risks based on how many devices are connected [3].

It's very popular to transfer images or videos utilizing multimedia, so securing multimedia content is crucial. One of the most popular methods for detecting motion estimation between frames in digital videos is motion vectors. Motion vectors, or the encoding of motion between frames, can be done without completely encoding each frame. predictive frame (P-frames), bi-directional (B-frames), and key frames—also referred to as reference frames or intra-frames—are the three categories into which each frame is divided. The projected inter-frames from another frame are called P-frames and B-frames [4]. Initially, the motion is assessed and sent on. The temporal context (with prediction being a type of context in the residual coding framework) is acquired through motion compensation, substituting handcrafted block-wise matching with optical flow-based pixel-wise warping. Ultimately, the frame is encoded and decoded with the help of temporal context. Most existing methods place greater emphasis on frame coding [5]. Most learnt image/video compression frameworks necessitate training distinct models for various rate-distortion sites, which raises the memory and training costs. P-frame is limited to referencing only previously encoded frames in the low-delay case, guaranteeing little delay. On the other hand, l reference is permitted for the current frame B-frame in the random-access scenario, which trades latency for increased compression efficiency [6]. Techniques for confirming integrity and authenticity must be updated frequently to reflect emerging trends in image processing and media manipulation, which are the driving force behind the creation of instruments that can precisely identify changes in photos and films [7].

An approach based on cryptography that uses methods such as AES, data encryption standard (DES), elliptic curve cryptography (ECC), and Rivest–Shamir–Adleman (RSA) to transform plain text into ciphertext. ECC includes minimal memory requirements, low computing power requirements, moderate network connectivity, and low communication bandwidth capability [8]. Numerous methods for protecting text data have been created, utilizing a variety of transformations such as wavelet transformations, wavelet transformations based on max-plus algebra, and various mathematical structures like elliptic curves [9]. The ElGamal Signature serves as the foundation for the elliptic curve digital signature technique, which is one of the cryptography applications in use. The legitimacy of communication is the outcome of this algorithm [10]. Simple encryption systems encode texts using symmetric keys and permutation matrices, and XOR, allowing for variable key sizes [11]. Since both the sender and the recipient are using similar technology, changing the key is the primary justification for utilizing elliptic curves for secure short message service (SMS). This guarantees thorough verification and the decryption of the SMS. At the precise moment of transmission, it is

thereby shielded against later attempts by others to decrypt it. ECC can be effectively utilized in this manner to secure and validate the correspondence between the two parties [12].

The AES algorithm is a type of encryption that employs symmetric keys and confidentiality licenses. An effective algorithm makes encryption unexpected, making it impossible to deconstruct using any technique [13]. Instead of using bits for all its calculations, AES uses bytes. As a result, AES interprets a plaintext block's 128 bits as 16 bytes. For matrix processing, these 16 bytes are organized in four rows and four columns. The length of the key determines how many rounds AES uses. AES employs 10 rounds for keys that are 128 bits, 12 rounds for keys that are 192 bits, and 14 rounds for keys that are 256 bits [14]. The Bruce attack on the AES 256-bit system has a key space of 2^{256}, which is a very large value [15]. The AES, known as Rijndael, is a well-known symmetric block cipher algorithm adopted by the United States government as a national encryption algorithm, and it provides portability, robustness, and high-level security against many cryptographic attacks. To have better performance, certain efforts have already been made in redesigning and reconstructing the AES algorithm [16].

The main contributions of the proposed method can be summarized as follows:

- Proposed an innovative, cutting-edge technique for text encryption and decryption that utilizes MPEG motion vectors.
- Generate both private and public keys for the sender and the recipient using the ECDH concept to generate private and public keys for both the sender and the recipient, and AES-256-bit encryption and decryption.
- The motion vector extraction, private and public key generation, and encryption and decryption times are calculated for the suggested method.
- Various message lengths are used to measure MSE, PSNR, CC, AE, and CR between plaintext and ciphertext.
- Comparison of the suggested work based on encryption and decryption execution times.
- Deployment the secure transmission of sensitive user instructions to an IoT endpoint.

The rest of this paper is structured as follows: Section 2 presents preliminaries about IoT systems, motion vectors, the ECDH concept, and the AES technique. Section 3 presented the related work of ECDH and AES. In Section 4, the proposed method algorithms are presented. The test and experimental results are provided in Section 5. The methodology framework is discussed in Section 6, while the paper's conclusion and future work are outlined in Section 6.

2 Preliminaries

2.1 IoT Systems

IoT systems have expanded quickly and are being utilized by a wide range of businesses and individuals in several industries, including industry and healthcare. IoT systems must integrate a distributed and secure system and protect the data they collect, particularly with regard to availability and integrity [17]. To improve edge-based network stability, real-time data from IoT sensors should be accessible to distant devices with mutual trust and concealed patterns. Even so, edge networks offer dependable and prompt data transfer for wireless technologies, but they are susceptible to network intrusions and the compromise of private information because of unauthorized device association [18]. Attackers could readily obtain the data gathered from the sensors because the data transmission is done over an open channel. Additionally, the attackers might pose as the user node, sensor node, or gateway node and carry out different attacks such as guessing and repeat attacks [19]. Several contemporary standards and technologies have been used to address interoperability issues in the integration of heterogeneous services and IoT devices. This is the benefit of

using and deploying other tools in accordance with the standards. However, this makes the material that is already available easier to comprehend [20].

2.2 Motion Vectors

Multimedia video encryption algorithms handle motion vectors in two distinct ways: one straightforward approach is to apply full encryption to motion vectors without exception, using traditional text encryption algorithms [21]. MPEG-4, H.264/advanced video coding (AVC), and high efficiency video coding (HEVC) are suggested for minimizing the storage space required for videos. To reduce the data size, these codecs utilize the motion compensation technique, which relies on estimating motion from neighboring frames [22]. Video codecs divide a video into groups of pictures (GOPs) as shown in Fig. 1, each comprising an Intra frame (I-frame) followed by a series of inter frames (P-frames or B-frames) [23]. This section will demonstrate the structure of video frames.

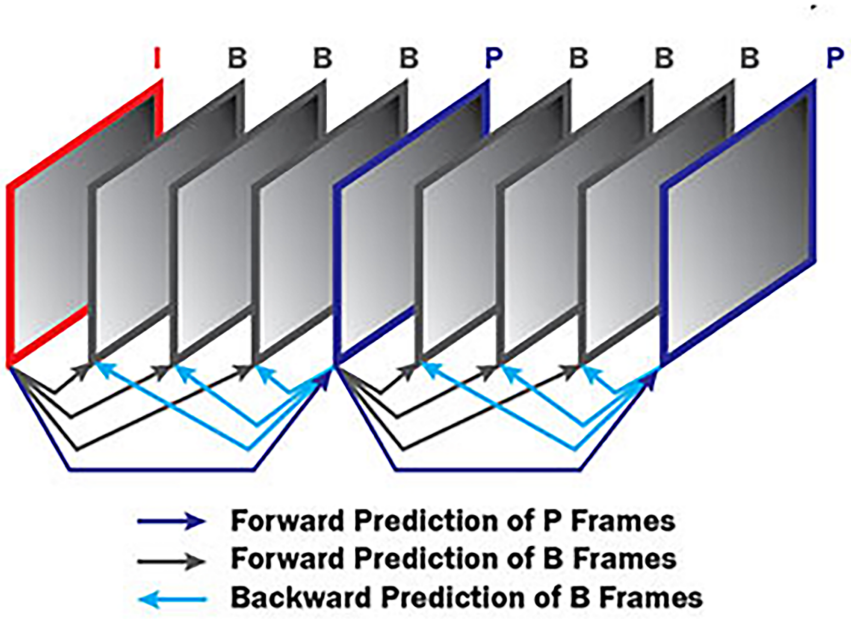

Figure 1: MPEG frames display order.

An MPEG video codec frame consists of different frame types including I-frames, P-frames, and B-frames, enabling efficient video compression.

i. Frame (Intra Frame):

Independently encoded key frames don't relate to other frames and don't need motion estimation. When recovering from packet loss during streaming, they act as key frames to guarantee error resilience. In addition to resetting motion vector dependencies to stop errors from spreading.

ii. P-Frame (Predictive Frame):

Encoded using motion-compensated prediction based on a prior reference frame (either an I-Frame or another P-Frame). Only save the differences from the reference frame. Include motion vectors that illustrate how pixel blocks (macroblocks) have shifted to the reference frame. Every motion vector indicates a corresponding block in the reference frame.

iii. B-Frame (Bi-Directional Frame):

Utilize both backward and forward frames as references for motion compensation. By interpolating motion from two directions, we can achieve the highest compression. Include pointing to blocks in two reference frames (backward and forward). A motion prediction is created by combining two motion vectors.

2.3 Optical Flow

Optical flow refers to the perceived movement of pixels across a series of frames. For optical flow to be estimated, the object's movement in the scene must correspond to a brightness displacement. The value of the intensity or brightness of the frame at location (*x, y*) at time *t* is represented by I (*x, y, t*) as shown in Fig. 2.

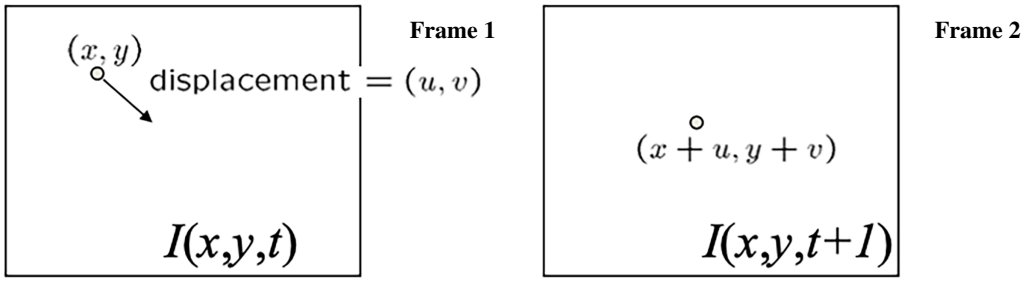

Figure 2: Motion detection from Frame 1 to Frame 2.

Consider a point at location (*x, y*) in the image at time *t*. This point moved to a new location (*x* + *u*, *y* + *v*) in the image at time *t* + *1*. Then, the brightness consistency equation will be:

$$I(x, y, t) = I(x + u, y + v, t + 1)$$

$$I(x + u, y + v, t + 1) = I(x, y, t) + uI_x + vI_y + I_t$$

$$I(x + u, y + v, t + 1) - I(x, y, t) = uI_x + vI_y + I_t$$

Let $I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}$, and $I_t = \frac{\partial I}{\partial t}$

$$uI_x + vI_y + I_t = 0$$

$$\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + I_t = 0$$

$$\nabla I \begin{bmatrix} u \\ v \end{bmatrix} + I_t = 0$$

Optical flow detects motion on a pixel-by-pixel basis, ensuring precise motion detection.

2.4 Elliptic Curve Diffie-Hellman

Over a finite field Fp, let Ep (a, b) be an equation for an elliptic curve. Ep (a, b) can be expressed as follows:

$$y^2 = x^3 + ax + b \ mod \ p \tag{1}$$

where *a*, *b* are two constants and *p* is a prime number that must be $4a^3 + 27b^2 mod p \neq 0$ [24]. ECC is used to create an asymmetric public key for two entities, each of which has its own private key. The two public keys

are then exchanged using the Diffie-Hellman key exchange (DHKE). There are EC public and private keys for each of these entities, and the public key will be the same [25].

Consider a generation point $G = (G_x, G_y)$, where G is a point on the elliptic curve. Parties A and B want to share keys to send a secret text message. To generate keys:

- **Part A:**

$$\text{Private key (PriA)} = \text{Randomly chosen integer } Pri_A \in [1, n-1] \tag{2}$$

where n is the generation point order $n \times G = \mathcal{O}$, $\mathcal{O}$ is the point at infinity.

$$\text{Public key (PubA)} = Pri_A \times G \tag{3}$$

- **Part B:**

$$\text{Private key (PriB)} = Pri_A \in [1, n-1] \tag{4}$$

$$\text{Public key (PubB)} = Pri_B \times G \tag{5}$$

Shared Secret Keys:

- **Part A:**

$$SA = Pri_A \times PubB \tag{6}$$

- **Part B:**

$$SB = PriB \times PubA \tag{7}$$

Both A and B are reached at the same shared secret point:

$$S = SA = SB \tag{8}$$

2.5 Advanced Encryption Standard

AES is regarded as the industry standard for encryption because it combines security, software, and hardware performance and flexibility [26]. AES is a form of block cipher with symmetric keys as depicted in Fig. 3. The three block ciphers that make up AES are AES-128, AES-192, and AES-256. Each cipher uses cryptography keys of 128, 192, and 256 bits, respectively, to encrypt and decrypt data in blocks of 128 bits [27]. The effectiveness of AES 256 encryption in thwarting brute-force attacks, differential cryptanalysis, and other cryptographic flaws has been demonstrated by numerous research studies that have thoroughly assessed its cryptographic strength and resilience [28]. For 128-bit keys, the AES system undergoes 10 rounds, 192-bit keys, 12 rounds, and for 256-bit keys, 14 rounds, either to retrieve the original plaintext or to deliver the final ciphertext [29].

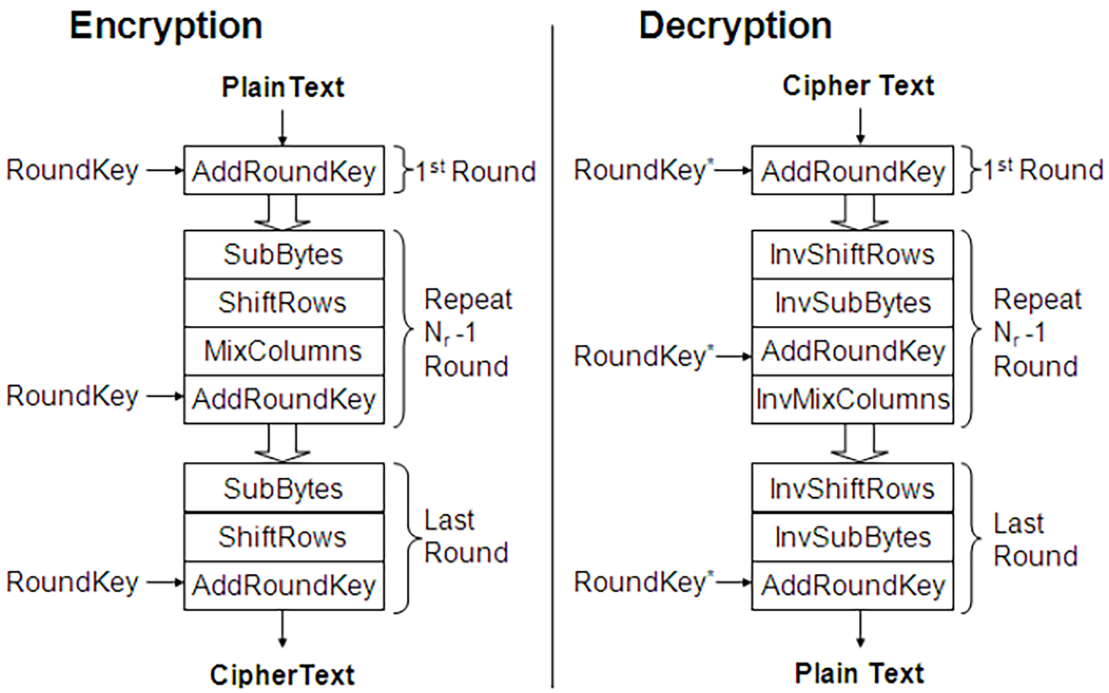

Figure 3: Advanced encryption standard technique.

3 Related Work

The protection of sensitive data is given the highest priority, motivating the execution of ongoing actions to prevent potential problems that may result from the processing of sensitive data by IoT devices and the constantly changing landscape of dangers [30]. The majority of IoT networks depend on network-based intrusion detection systems (IDS) to secure the network's collective nodes because most IoT devices lack sufficient processing and memory capabilities [31]. Deploy and run networks and systems in a way that reduces their determinism, homogeneity, and staticity to raise the cost of attacks. However, the core features of IoT systems, such energy efficiency and heterogeneity, necessitate defense tactics that concentrate their energies on lowering the likelihood that an assault will succeed [32].

The use of elliptic curve cryptography for text encryption is typically recognized due to its robustness against various attacks, stemming from the challenge of identifying the generation point. However, certain studies do not consider other crucial factors like time complexity and key space for the techniques employed. Although a secret system is used for banking, the ECC is based on a small key size, rendering this method insufficiently secure and unable to meet high security standards [33]. Time complexity is regarded as a crucial component of the cryptography concept. The authors in [34] employed ECDH to encrypt and decrypt secret text messages; nevertheless, the encryption and decryption complexity times are approximately 0.0016 and 0.008 s, respectively, for small text message lengths. When comparing the concepts of RSA and ECC based on key space size, the suggested method's ECC key space range is not specifically mentioned in [35]. To measure the encryption and decryption times for a 20 kilobyte (KB) file data to 0.015 and 0.019 s, respectively, the authors employ a hybrid strategy that smoothly combines blockchain technology with the radio frequency identifier (RFID) reader, generative adversarial neural system, and ECDH cryptographic techniques [36]. A brief text message of roughly 39 bytes was encrypted and decrypted by the authors in [37] using the Chinese remainder theorem, ECC points, which takes 0.0012 and 0.0016 s, respectively. The authors in [38] encrypted

and decrypted text messages using random access encryption specification (RAES) in 0.40 s by utilizing the rail fence cipher (RFC) and the AES. In [39], the authors employed AES to encrypt text communications in the cloud, achieving encryption and decryption times of 2.268 and 2.34 s, respectively, for 64 bytes. Moreover, encryption and decryption took $8.114 * 10^{-4}$ and $1.754 * 10^{-4}$ s, respectively, for 128 bytes. With a new XOR matrix 16 * 16 (S-box) value, the authors in [40] proposed a hyperchaotic system to obtain more secure AES with an average time of roughly 0.275 s. Using the Java programming language and the AES idea, the authors were able to attain encryption times of 40, 50, 60, 62, and 56 s, and decryption times of 37, 46, 57, 60, and 61 s, correspondingly, for 50, 100, 150, 200, and 500 bytes [41]. Using the LabVIEW 2016 simulation design, the authors in [42] determined the encryption and decryption timings for the AES algorithm and came up with values of 2.05, 3.606, 4.88, and 13.371 s for text file sizes of 9150, 538, 1180, and 3535 bytes, respectively. The authors in [43] have encrypted text files of sizes 20, 50, 75, 100, and 150 by using a mixture of AES, ECC, and secure hash algorithm (SHA)-256. Decryption times were 80, 82, 83, 88, 92, and 113 ms, and encryption timings were 49, 50, 57, 65, 71, and 88 ms.

4 Proposed Method

The proposed method begins with the division of a digital MPEG video into frames. The video size is 60 megabytes (MB) with dimensions [1080, 1920]. The same reference video is used for key generation because the video source is a generic pre-stored MPEG video file rather than real-time capture. This eliminates the need for IoT devices to have video recording capabilities. With no semantic connection to the IoT environment or communication content, the motion vectors generated from this MPEG video just function as a reproducible entropy source, guaranteeing the scheme's adaptability across a variety of IoT terminals. Key derivation becomes deterministic and eliminates the need for real-time video transmission or content coordination during communication when both parties independently process the same predefined video file frame sequence to accomplish synchronization. Once extracted, convert two successive frames from Red-Green-Blue (RGB) to grayscale so that motion vectors can be extracted. Then, motion vectors will be extracted along the x-axis and y-axis. Motion vectors (MV) extraction consumes about 1.2 to 1.5 s. The resulting motion vectors have a floating-point value (not integers) as they represent motion over a certain period. To map these values to an elliptic curve equation, the numbers must be converted to integers by multiplying them by a scale factor of 1000. Subsequently, the result will be converted to a 1D column vector, which undergoes an XOR operation, and then a hash function to map it onto the EC for generating private, public, and shared secret keys. The sender's private key is generated from the resulting XOR operation. An initialization vector (IV) is created, and cryptography processes are done, both AES and IV to encrypt and decrypt confidential text messages.

4.1 Method Framework

The proposed method relies on an initial step and two additional steps. The first fundamental step involves extracting motion vectors from two successive video frames. The high definition (HD) quality video that is being used has dimensions of [1080, 1920], and 2,073,600 points is the total number of points. After extraction, the motion vectors that are produced are floating points, so to ensure integer values, these values must be scaled to a large value. When the scale factor is N, where N >= 1000, integer values are the outcome. Following their conversion to a 1D column vector, the motion vectors over the x and y axes are represented by the values d_x = [2,073,600, 1] and d_y = [1, 2,073,600]. An XOR procedure is used to get a unique value for dx and dy, then a total XOR result is obtained (d_{xy}). The first private key is then immediately derived by mapping the resultant value (d_{xy}) to the elliptic curve equation using a hash function, SHA-256 to prevent entropy loss [44].

Using the parameters, a = −40, b = 80, and p = 997, consider the elliptic curve equation: $y^2 = x^3 + ax + b \bmod p$. A total of 1005 points were generated as shown in Fig. 4. The selected generation point is $(G_x, G_y) = (439, 791)$, which is placed on the elliptic curve. The two successive video frames also contain this point. Using the elliptic curve Diffie-Hellman Exchange principle, Alice and Bob can create private and public keys in the following way:

Alice's private key = $d_{xy} * \bmod (p - 1) + 1$

Alice's public key = (Alice's private key * G_x, Alice's private key * G_y) mod p

Bob's private key = random number ([1, (p − 1)]

Bob's public key = (Bob's private key * G_x * δ_x + Bob's private key * G_y * δ_y) mod p,

where $\delta_x + \delta_y = 1$.

To compute the shared keys:

Shared Key = (Alice's private key * Bob's public key * δ_A + Bob's private key * Alice's public key * δ_B) mod p,

where $\delta_A + \delta_B = 1$.

Next, either Alice's or Bob's shared key is used to construct the AES-256 which is generated from the shared secret key. An AES-256 key and an IV are created from random values to produce the ciphertext, which ranges from [0, 255]. IV is a random or pseudo-random value used to enhance security in encryption algorithms by preventing repetition:

- Creating a unique value in each encryption step is used for the encryption process.
- Various IVs are generated to ensure that different ciphertexts are generated, even if the same plaintext is encrypted more than once.
- IV sets up the AES counter block, which generates the key by incrementing with each AES block encryption.

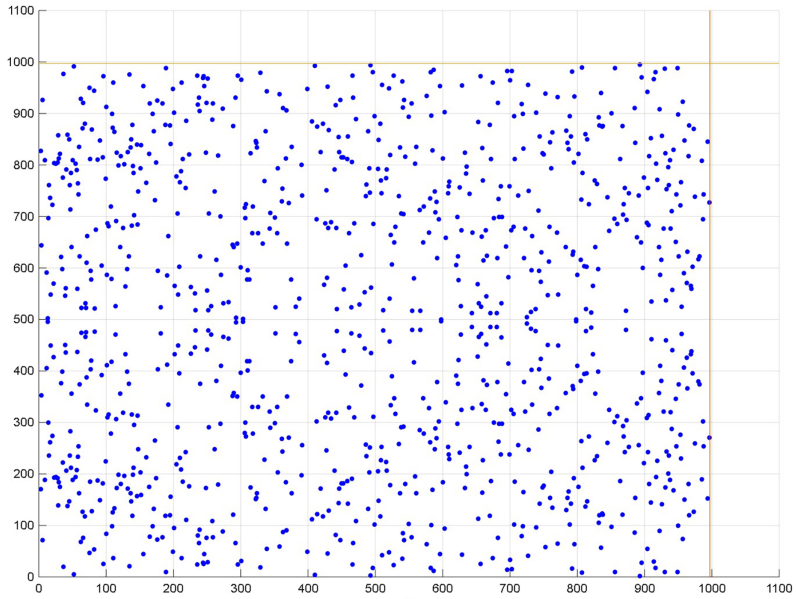

Figure 4: ECC generated points.

The encrypted text will be generated once the encryption procedure is complete and the decrypted is aware of which IV was used. The same IV is then used to initialize the ciphertext for the decryption process using AES, as shown in Fig. 5.

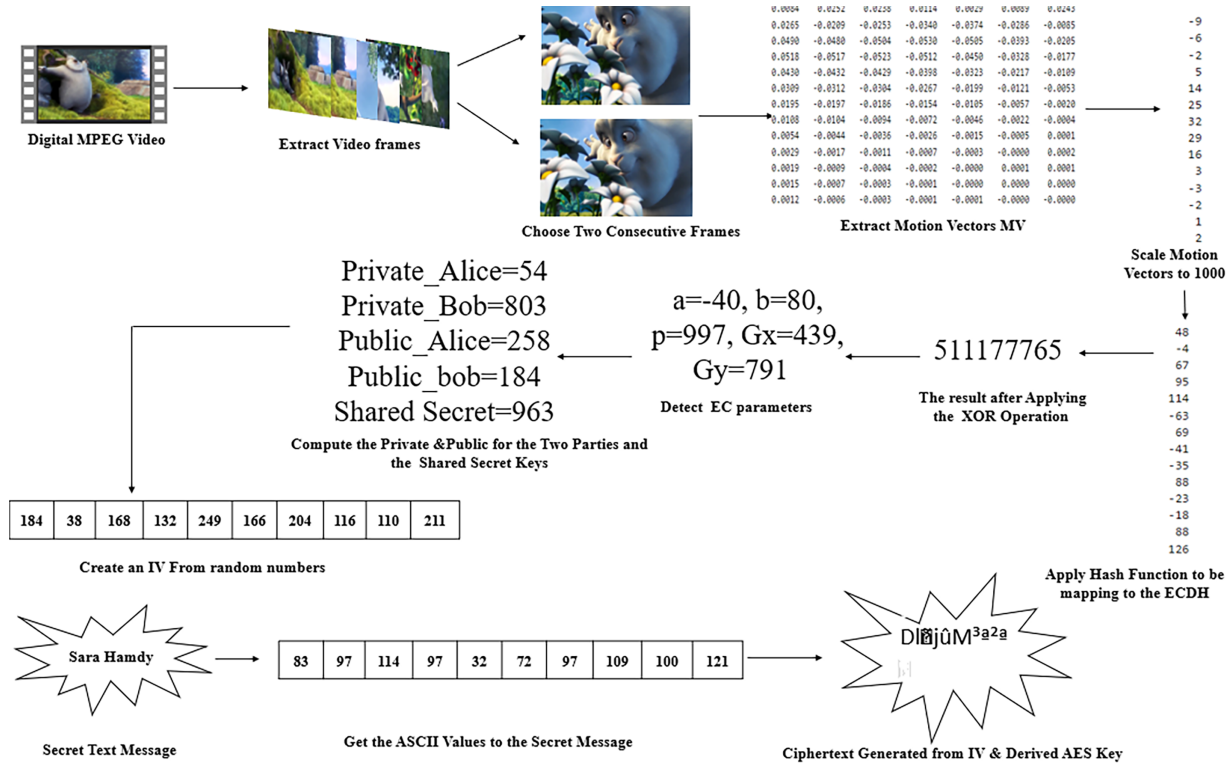

Figure 5: General proposed work architecture.

4.2 Proposed Method Algorithm

The algorithm for the proposed method comprises three parties: First, motion-based key generation, which relies on the mapping of motion vectors to the creation of shared, public, and private secret keys. The second step is the encryption procedure, which involves encrypting text using the created key, or shared secret key, that is obtained from Algorithm 1. The third step is the decryption process, which deals with getting the original text.

Algorithm 1: Motion-based key generation

Input: MPEG video file.
Output: private/public key pair, shared secret key.
1. Obtain an MPEG digital video.
2. Split the video into frames.
3. Select two consecutive frames.
4. Using optical flow, extract the motion vectors (d_x, d_y).
5. Apply a scale factor of N to scale d_x, d_y.
6. Convert the previous factor to a 1D column.
7. Get the XOR procedure of the resulting factor (d_{xy}).
8. Utilizing the ECDH principle to generate private and public keys.

(Continued)

Algorithm 1 (continued)

P: prime number
G_x, G_y: generator coordinates
δ_x, δ_y: weights, where $\delta_x + \delta_y = 1$
δ_A, δ_B: weights, where $\delta_A + \delta_B = 1$
Sender_Private key = (d_{xy} mod (p − 1))
Sender_public key = ((Sender Private key * G_x) mod p, (Sender_Private key * G_y))
Reciever_private key = random value (1, p − 1)
Reciever_public key = Sender_Private key * (G_x * δ_x) mod p
Shared secret_key = ((Sender_Private key * Reciever_public key) * δ_A) +
((Reciever_private key * Sender_public key * δ_B) mod p
9. Obtain the private, public, and shared secret keys.

The algorithm's first part starts by using optical flow to extract motion vectors (d_x, d_y) from two consecutive video frames, scales them by a factor N, and transforms them into a one-dimensional column vector (d_{xy}), which is then transformed via an XOR procedure to serve as a seed for cryptographic key generation. Next, it applies a modified version of the ECDH key exchange, where the sender's private key is derived from d_{xy} modulo (p − 1). Finally, it uses a weighted combination of generator point coordinates (G_x, G_y) with weights δ_A and δ_B for key contribution as represented in Algorithm 1. For the practical video size, the time complexity of Algorithm 1 reduces to *O(n)*. The primary expense is optical flow computation, which takes linear time in relation to the number of pixels (*n* = width × height) by processing each pixel in the two chosen frames individually.

To ensure cryptographic randomness for every encryption operation, the second algorithm first converts a text message into its corresponding american standard code for information interchange (ASCII) values before creating a random IV between 0 and 255. A 256-bit AES key is derived from the derived shared secret key. The AES key, along with the IV, is used in a symmetric encryption process to encrypt the ASCII representation of the message, ultimately producing the final ciphertext output as shown in Algorithm 2. The total time complexity of Algorithm 2 is *O(n)*. Because the execution time halves when the message length doubles, processing *n* characters confirms *O(n)* complexity.

Algorithm 2: Encryption process

Input: text message, shared secret key.
Output: ciphertext
1. Receive an original text message.
2. For each text message, do:
   a. Convert text message to ASCII value.
   b. Generate random IV in range [0, 255].
   c. Derive 256-bit AES key from the shared secret key.
   d. Use IV, AES key to encrypt the text message.
3. Obtain the ciphertext.

The time complexity of Algorithm 3 is *O(n)*, where *n* is the length of the ciphertext in bytes. By re-deriving the identical 256-bit AES key from the same shared secret key used during encryption, this decryption algorithm reverses the encryption process. It then uses this key and the transmitted IV to perform symmetric decryption on the received ciphertext, reversing the cryptographic transformation to recover the

original sequence of ASCII values, which are then converted back into their corresponding characters to faithfully reconstruct and obtain the original plaintext message as shown in Algorithm 3.

Algorithm 3: Decryption Process

Input: ciphertext (encrypted text message)
Output: Original text message (decrypted message)
1. Re-derive 256-bit AES key from the shared secret key.
2. Use IV and AES key to decrypt the ciphertext.
3. Convert the resulting ASCII values back to text.
4. Obtain the original text.

5 Experimental Results

The application of the proposed approach is discussed in this section. MATLAB 2025a has been used to implement the proposed method, utilizing 60 MB of HD-quality MPEG footage. After extraction, the video used in this method contains 2932 frames. Different executive video frames have been used for each text message compared to other text messages. The following measures [33,34], including MSE, PSNR, CC, AE, and CR, were used to evaluate the suggested approach:

$$MSE_x = \frac{1}{N}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[S(i,j) - R(i,j)]^2, \tag{9}$$

where N is equal to $m * n$, where n is the number of columns in the frame and m is the number of rows. S is the value of the encrypted message, and R is the decimal representation of the original secret message. The following is the definition of PSNR:

$$PSNR = 10 * log_{10} * \frac{(MAX_I)^2}{MSE_t}, \tag{10}$$

where MAX, which equals 255, is the maximum pixel value in a specific video frame. This is how the CC is calculated:

$$CC = \frac{\sum (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum (x_i - \overline{x})^2 (y_i - \overline{y})^2}}, \tag{11}$$

where $\boldsymbol{x_i}$ is the original message value, $\overline{\boldsymbol{x}}$ is the mean of $\boldsymbol{x}$, $\boldsymbol{y_i}$ is the value of the encrypted message and $\overline{\boldsymbol{y}}$ is the mean of $\boldsymbol{y}$. The AE is computed as:

$$\text{AE} = \frac{\text{Number of Changed bits in Ciphertext}}{\text{Number of bits in Ciphertext}} \tag{12}$$

Compression ratio is computed as the size of the ciphertext/the size of the plaintext. When comparing the ciphertext and the plaintext, the result must be close to zero, low in PSNR, and extremely high in MSE. For all calculations between the sizes of the ciphertext and plaintext, the compression ratio must be 1, and the AE must be more than or equal to 45. Thousands of seconds must be the ideal execution time. The execution timings for creating private and public keys, as well as the encryption and decryption times for the proposed method, as shown in Table 1.

Table 1: Proposed method private, public key generation, encryption, and decryption times.

Message Size (bytes)	Private & Public Key Generation (seconds)	Encryption Time (seconds)	Decryption Time (seconds)
10	0.001702	0.002136	0.000681
20	0.001470	0.001471	0.000632
30	0.001614	0.001529	0.000633
50	0.001564	0.001514	0.000648
75	0.001572	0.001502	0.000635
80	0.001491	0.001527	0.000610
100	0.001492	0.001447	0.000650
150	0.001466	0.001533	0.000682
175	0.001883	0.001424	0.000671
200	0.001528	0.001448	0.000637
250	0.001483	0.001606	0.000679
300	0.001468	0.001418	0.000633

In lightweight cryptographic time calculations, the measured error margins of 0.0359 ms for encryption and 0.0141 ms for decryption, obtained for a 10-byte message length, demonstrate exceptional precision. These margins amount to relative errors far below 5%, demonstrating extremely dependable and repeatable timing measurements, since encryption times for software implementations on resource-constrained systems typically range from 1 to 5 ms. The consistency of the cryptographic procedures and the statistical robustness of the proposed performance statistics are confirmed by such narrow confidence intervals.

Using two consecutive video frames in each text length, the time measurement is applied to messages ranging in size from 10 to 300 bytes, resulting in varying MV values. The encryption and decryption processes, as well as the production of private and public keys, take milliseconds. These activities cannot affect energy consumption, which is correlated with operating times, control processing unit (CPU) speed, and message size. The proposed lightweight appears to have achieved a high level of security based on the metrics MSE, PSNR, CC, and AE values, which are deduced indirectly from the quality of the original and encrypted text. Since the original material is less identifiable, a high MSE and low PSNR suggest significant distortion between the original and processed text, which may signal higher confidentiality. There is no linear statistical relationship between the original message and ciphertext if the CC is close to 0. This is precisely what a safe cipher should accomplish: it should make ciphertext seem random and unconnected to plaintext, making it impossible for attackers to identify patterns. AE is equals and greater than 45, which shows that the algorithm is highly sensitive to changes in input and successfully masks statistical correlations. The values of the MSE, PSNR, CC, and AE are shown in Table 2.

The side-channel attack, which demonstrates that the recovered keys' CC is close to zero, indicates that the actual key was not properly extracted from the data. This is another secure implementation. The idea that a significant key recovery was not achieved by the assault is supported by a correlation close to 0, which shows that there is no visible relationship between the guesses and the real key as shown in Fig. 6. The findings point to significant weaknesses in the side-channel measurements used to retrieve the key, which may point to a robust defense mechanism in the cryptographic implementation. Either the cryptographic technology is resistant to these types of attacks, or the side-channel data was insufficient to distinguish between multiple significant competitors.

Table 2: MSE, PSNR, CC, and AE values for the proposed method.

Message Length (bytes)	MSE	PSNR	CC	AE
10	1521	16.31	−0.39604	57
20	13,225	6.92	0.0355	45
30	5476	10.75	−0.232	45
50	7396	9.44	−0.02791	50
75	9216	8.49	0.0728	46
80	9409	8.4	0.05417	47
100	10,201	8.04	−0.00762	47
150	13,456	6.84	−0.00967	46
175	10,000	8.13	0.04928	47
200	10,201	8.04	0.013814	46
250	10,000	8.13	−0.08518	48
300	13,456	6.84	−0.036662	46

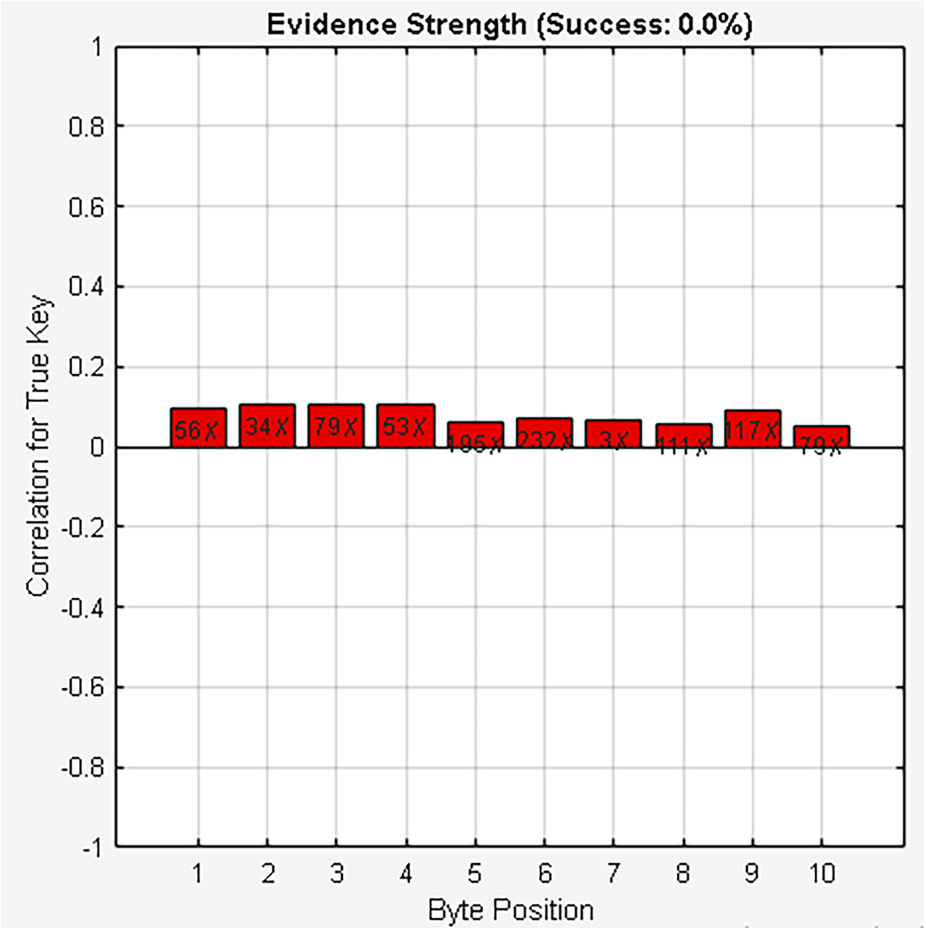

Figure 6: Side-channel attack.

The proposed method has been performed on large video frame sizes since larger frame dimensions produce more points and require the selection of appropriate EC parameters to produce a high key space.

[1080, 1920] is the size of the video frames used in this method; the total video frame points are 2.073.600, and the points produced by the EC parameters are 1005 points. The greatest points will be rewarded on the EC if the EC parameters are increased. The selected generation point has a subgroup order equal to 1005. In addition to using the AES-256-bit algorithm, the IV was created using random numbers in the range [0, 255].

Based on these values, the key space of the suggested method is 2^{256}, or $1.1579 * 10^{77}$ possible keys that are thought to be impenetrable to the most popular attack, the brute force attack, which breaks the key and reveals the secret message while it is being transmitted over insecure channels. There are some considerations about using the proposed method:

- The number of the elliptic curve points must be appropriate for the number of frame points.
- Reducing the frame dimensions will lead to a lower number of points, and some of these points may not be allocable on the elliptic curve.

If a user decides to send an instruction to an IoT device "Air Conditioning", the secret instruction is: "change temperature to 28°C" which can be securely transmitted to this IoT endpoint device. Before being transmitted across the network, the plaintext instruction is encrypted to prevent unwanted access or alteration. This resultant ciphertext, incomprehensible data string that has no significance. This encrypted instruction would be all that a malevolent actor (a hacker) could see if they were to intercept this confidential communication. The user's purpose and the system's security are safeguarded by this cryptographic layer, which guarantees data confidentiality and integrity because the attacker cannot interpret the original instruction or change it in a meaningful way without being discovered as shown in Fig. 7.

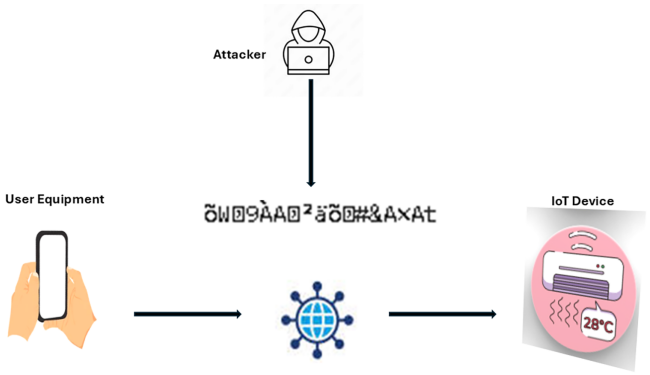

Figure 7: Secret instruction transmission from user's mobile to an IoT device

6 Discussion

A modern lightweight cryptographic framework has been developed for secure IoT instruction transmission by exploiting the motion vectors extracted from MPEG video frames as a high-entropy source. In the proposed scheme, these motion vectors are first processed and mapped onto the ECDH key agreement framework, where they contribute to the derivation of the private keys, public keys, and the resulting shared secret key, as conceptually depicted in Fig. 5. This shared secret is then combined with an IV to derive the AES session key, which is subsequently used to perform efficient symmetric encryption and decryption of IoT control messages. Because the entire procedure operates on the order of milliseconds, the method is particularly suitable for resource-constrained platforms such as embedded devices, low-power IoT nodes, and smartphones, as well as real-time video streaming services where latency and throughput constraints are stringent.

A practical IoT use case involving secret instruction transmission is presented in Fig. 7, demonstrating how the proposed cryptographic pipeline can be integrated into a realistic communication scenario. Table 1 reports the optimal parameter settings and associated key generation, encryption, and decryption times, highlighting that the scheme maintains strong security while imposing minimal computational overhead. To quantify robustness, Table 2 presents a comprehensive security analysis based on MSE, PSNR, CC, and AE. The obtained results indicate a high resistance to statistical, differential, and brute-force attacks, confirming that the method provides a strong cryptographic defense against a wide range of adversarial strategies.

To further validate performance and scalability, the proposed technique was evaluated over varying message lengths, from 10 bytes up to 10,000 bytes, with encryption and decryption times summarized in Table 3. The timing results show nearly linear growth with message size while remaining within an acceptable range for real-time and interactive IoT applications. Moreover, a fidelity analysis between the original and decrypted messages yields MSE = 0, PSNR = ∞, and CC = 1, demonstrating that the decryption process is perfectly reversible and that no distortion or information loss is introduced by the cryptographic operations. To demonstrate the proposed method's resilience to attacks, a second comparison was made using MSE, PSNR, and CC measures as shown in Table 4. A side channel attack has been carried out as demonstrated in Fig. 6. The proposed method has a protective defense against the assault by obtaining minimal values of CC, indicating that the attack is unable to extract the used key. Taken together, these findings underscore that the proposed motion-vector–driven ECDH–AES framework constitutes an efficient, scalable, and highly secure solution for protecting sensitive text instructions in heterogeneous IoT environments as represented in Fig. 7.

Table 3: Comparison between other related work according to encryption and decryption times.

Reference Number	No. of Characters	Encryption Time		Decryption Time	
		Ref. No.	Proposed	Ref. No.	Proposed
Reference [45]	100	0.0409	0.001447	0.0343	0.000650
	200	0.0554	0.001448	0.00349	0.000637
	300	0.0572	0.001418	0.00367	0.000633
	400	0.0623	0.001023	0.00382	0.000871
	500	0.0692	0.001073	0.0041	0.000739
	600	0.0735	0.001004	0.00421	0.000688
	700	0.0812	0.001059	0.00478	0.000698
	800	0.0869	0.001767	0.00511	0.000821
	900	0.0967	0.001514	0.00565	0.000775
	1000	1.0013	0.001032	0.00621	0.000706
Reference [46]	300	0.0125	0.001418	0.0084	0.000633
	1000	0.0171	0.001032	0.0142	0.000706
	2000	0.0236	0.001055	0.0194	0.000739
	5000	0.0272	0.001077	0.0232	0.000749
	10,000	0.0388	0.001303	0.0327	0.000957
Reference [47]	374	0.095	0.001016	0.0053	0.000688
	748	0.099	0.001013	0.049	0.000688
	1122	0.111	0.001113	0.051	0.000709
	2244	0.112	0.001058	0.082	0.000760

Table 4: Comparison between other related work according to MSE, PSNR, and CC measurements.

Reference Number	Message Length	MSE		PSNR		CC	
		Ref. No.	Proposed	Ref. No.	Proposed	Ref. No.	Proposed
Reference [48]	10	8370	10,641	18.343	8.48	−0.1406	−0.44
	20	13,923	14,641	15.1759	6.48	−0.3652	−0.3
	30	8321	1024	20.3225	18.03	0.0833	−0.04
	50	10,414	2025	17.598	15.07	−0.0155	−0.06
Reference [49]	24	7121	12,100	22.11	7.3	0.026	0.025
	48	11,277	9404	17.52	8.4	−0.3739	−0.18
	120	8063	12,002	20.875	7.05	0.0493	−0.0082
	144	4940	8612	25.77	6.32	0.3281	−0.08
	240	7689	10,100	21.35	8.02	−0.1261	−0.07

7 Conclusion

Using a combination of ECDH and AES approaches, this work has presented a contemporary cryptographic method created using MPEG motion vectors. After scaling the horizontal and vertical motion vectors by 1000, the proposed method generates private and public keys from the resulting XOR of these values. The text message is then encrypted and decoded using the AES-256 technique after an IV is created using random values in the [0, 255] range. The proposed approach started with 10 bytes and compares to 10,000 bytes, yielding perfect results, based on key generations, encryption and decryption timings, and resistance attack metrics MSE, PSNR, CC, AE, and CR. Empirical deployment demonstrates the methodology's capacity to convey encrypted administrative instructions from a unique user interface to a unique IoT endpoint, ensuring data integrity throughout the transmission process. A limitation of this work is that it is not appropriate to use a small video frame's dimension for implementation, since it generates a limited number of points, and dealing with the elliptic curve may not locate some of these points. The suggested system exhibits limited scalability in transferring instructions from user devices to IoT devices, as each device pairing requires a distinct video-based key exchange mechanism. This requirement leads to significant management and operational overhead, rendering the approach impractical as the network size expands. Furthermore, synchronization challenges arise when multiple users issue commands concurrently, resulting in increased contention, coordination complexity, and potential degradation of system performance. Future work will employ the block matching technique, an alternative motion vector estimation approach, to implement a comparable strategy and evaluate its effectiveness against the current method in terms of accuracy, computational complexity, and robustness across diverse video sequences.

Acknowledgement: The authors would like to thank the Deanship of Scientific Research at Shaqra University for supporting this work.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Sara H. Elsayed, conceptualization, software, writing and preparation; Rodaina Abdelsalam, data curation and investigation; Mahmoud A. Ismail Shoman, resources and project administration; Raed Alotaibi, writing and visualization; Omar Reyad, idea proposal, review, editing and supervision. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The video file that supports the findings of this research is publicly available as indicated in reference [50].

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dubey K, Sharma SC, Kumar M. A secure IoT applications allocation framework for integrated fog-cloud environment. J Grid Comput. 2022;20(1):5. doi:10.1007/s10723-021-09591-x.
2. Fernandez EB, Washizaki H, Yoshioka N, Okubo T. The design of secure IoT applications using patterns: state of the art and directions for research. Internet Things. 2021;15:100408. doi:10.1016/j.iot.2021.100408.
3. Ragesh GK, Kumar A. Trust-based secure routing and message delivery protocol for signal processing attacks in IoT applications. J Supercomput. 2023;79(3):2882–909. doi:10.1007/s11227-022-04766-z.
4. Turner R, Banerjee N, Banerjee S. Using video motion vectors for structure from motion 3D reconstruction. In: Proceedings of the 19th International Conference on Signal Processing and Multimedia Applications; 2022 Jul 14–16; Lisbon, Portugal. p. 13–22. doi:10.5220/0011263600003289.
5. Qi L, Li J, Li B, Li H, Lu Y. Motion information propagation for neural video compression. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2023 Jun 17–24; Vancouver, BC, Canada. p. 6111–20. doi:10.1109/CVPR52729.2023.00592.
6. Çetin E, Yılmaz MA, Tekalp AM. Flexible-rate learned hierarchical bi-directional video compression with motion refinement and frame-level bit allocation. In: 2022 IEEE International Conference on Image Processing (ICIP); 2022 Oct 16–19; Bordeaux, France. p. 1206–10. doi:10.1109/ICIP46576.2022.9897455.
7. Sheng X, Chen P, Wang M, Zhang L, Wang S, Wu D. Fine-grained motion compression and selective temporal fusion for neural B-frame video coding. arXiv:2506.07709. 2025.
8. González Fernández E, Sandoval Orozco AL, García Villalba LJ. Digital video manipulation detection technique based on compression algorithms. IEEE Trans Intell Transp Syst. 2022;23(3):2596–605. doi:10.1109/TITS.2021.3132227.
9. Hureib ES, Gutub AA. Enhancing medical data security via combining elliptic curve cryptography with 1–LSB and 2–LSB image steganography. Int J Comput Sci Netw Secur. 2020;20(12):232–41.
10. Sattar KA, Haider T, Hayat U, Bustamante MD. An efficient and secure cryptographic algorithm using elliptic curves and max-plus algebra-based wavelet transform. Appl Sci. 2023;13(14):8385. doi:10.3390/app13148385.
11. Juhari J, Andrean MF. On the application of noiseless steganography and elliptic curves cryptography digital signature algorithm methods in securing text messages. CAUCHY J Mat Murni Dan Aplikasi. 2022;7(3):483–92. doi:10.18860/ca.v7i3.17358.
12. Kinganga J, Kasoro N, Musesa A. Dynamics data encryption based on chaotic functions and elliptic curves: application to text data. Al-Mustansiriyah J Sci. 2025;36(1):56–68. doi:10.23851/mjs.v36i1.1616.
13. Nibigira N, Havyarimana V, Xiao Z. Sensitive information security based on elliptic curves. World J Eng Technol. 2024;12(2):274–85. doi:10.4236/wjet.2024.122018.
14. Laurentinus, Pradana HA, Sylfania DY, Juniawan FP. Improving the SMS security and data capacity using advanced encryption standard and Huffman compression. In: Proceedings of the Sriwijaya International Conference on Information Technology and Its Applications (SICONIAN 2019); 2019 Nov 16; Palembang, Indonesia. p. 194–202. doi:10.2991/aisr.k.200424.028.
15. Wuttidittachotti P, Natho P. Improved ciphertext-policy time using short elliptic curve Diffie–Hellman. Int J Electr Comput Eng. 2023;13(4):4547. doi:10.11591/ijece.v13i4.pp4547-4556.
16. Aikins-Bekoe S, Ben J. Elliptic curve diffie-Hellman (ECDH) analogy for secured wireless sensor networks. Int J Comput Appl. 2020;176(10):1–8. doi:10.5120/ijca2020920015.
17. Alrubei SM, Ball E, Rigelsford JM. A secure blockchain platform for supporting AI-enabled IoT applications at the edge layer. IEEE Access. 2022;10:18583–95. doi:10.1109/ACCESS.2022.3151370.

18. Elhoseny M, Siraj M, Haseeb K, Nawaz M, Altamimi M, Alghamdi MI. Energy-efficient mobile agent protocol for secure IoT sustainable applications. Sustainability. 2022;14(14):8960. doi:10.3390/su14148960.
19. Vinoth R, Deborah LJ. An efficient key agreement and authentication protocol for secure communication in industrial IoT applications. J Ambient Intell Humaniz Comput. 2023;14(3):1431–43. doi:10.1007/s12652-021-03167-z.
20. Alberternst S, Anisimov A, Antakli A, Duppe B, Hoffmann H, Meiser M, et al. Orchestrating heterogeneous devices and AI services as virtual sensors for secure cloud-based IoT applications. Sensors. 2021;21(22):7509. doi:10.3390/s21227509.
21. Irshad RR, Hussain Z, Hussain I, Hussain S, Asghar E, Alwayle IM, et al. Enhancing cloud-based inventory management: a hybrid blockchain approach with generative adversarial network and elliptic curve diffie helman techniques. IEEE Access. 2024;12:25917–32. doi:10.1109/ACCESS.2024.3367445.
22. Menandas J, Christo MS. Chinese remainder theorem-based encoding of text to point elliptic curve cryptography. J Adv Res Appl Sci Eng Technol. 2024;47(2):148159. doi:10.37934/araset.47.2.148159.
23. Hussain HS, Yahya S, Jaimun M, Khidzir NZ. A hybrid approach for a secured information security using modified encryption technique. Int J Multidiscip Res. 2022;4(6):1124. doi:10.36948/ijfmr.2022.v04i06.1124.
24. Rani R, Bathla RK. Performance of secure framework AES algorithm using cloud computing. J Sci Res. 2024;16(2):381–403. doi:10.3329/jsr.v16i2.66682.
25. Zghair HK, Manaa ME, AL-Murieb SSA, Abd Al-Razaq FJ. Analysis and description S-box generation for the AES algorithm-a new 3D hyperchaotic system. Bulletin EEI. 2023;12(3):1639–47. doi:10.11591/eei.v12i3.4824.
26. Ogundoyin IK, Ogunbiyi DT, Adebanji S, Okeyode YO. Comparative analysis and performance evaluation of cryptographic algorithms. Uniosun J Eng Environ Sci. 2025;4(1):39–47. doi:10.36108/ujees/2202.40.0140.
27. Latif IH. Time evaluation of different cryptography algorithms using LabVIEW. IOP Conf Ser Mater Sci Eng. 2020;745(1):012039. doi:10.1088/1757-899X/745/1/012039.
28. Liu Z, Li X. Motion vector encryption in multimedia streaming. In: 10th International Multimedia Modelling Conference 2004; 2004 Jan 5–7; Brisbane, QLD, Australia. p. 64–71. doi:10.1109/MULMM.2004.1264968.
29. Han H, Yang G, Huo Y, Lu Z, Wen JR. Complex action segmentation in compressed videos. In: 2021 IEEE International Conference on Multimedia and Expo (ICME); 2021 Jul 5–9; Shenzhen, China. p. 1–6. doi:10.1109/ICME51207.2021.9428317.
30. Ullah I, Noor A, Nazir S, Ali F, Ghadi YY, Aslam N. Protecting IoT devices from security attacks using effective decision-making strategy of appropriate features. J Supercomput. 2024;80(5):5870–99. doi:10.1007/s11227-023-05685-3.
31. Osei A, Al Mtawa Y, Halabi T. Mitigating adversarial reconnaissance in IoT anomaly detection systems: a moving target defense approach based on reinforcement learning. EAI Endorsed Trans IoT. 2024;10:1–14. doi:10.4108/eetiot.6574.
32. Mercado-Velázquez AA, Escamilla-Ambrosio PJ, Ortiz-Rodríguez F. A moving target defense strategy for Internet of Things cybersecurity. IEEE Access. 2021;9:118406–18. doi:10.1109/ACCESS.2021.3107403.
33. Hadi HH, Ali Neamah A. An image encryption method based on modified elliptic curve Diffie-Hellman key exchange protocol and Hill Cipher. Open Eng. 2024;14(1):20220552. doi:10.1515/eng-2022-0552.
34. Moussa KH, El-Sakka AH, Shaaban S, Kheirallah HN. Group security authentication and key agreement protocol built by elliptic curve diffie Hellman key exchange for LTE military grade communication. IEEE Access. 2022;10:80352–64. doi:10.1109/ACCESS.2022.3195304.
35. Gamido HV. Implementation of a bit permutation-based advanced encryption standard for securing text and image files. Indones J Electr Eng Comput Sci. 2020;19(3):1596. doi:10.11591/ijeecs.v19.i3.pp1596-1601.
36. Mulya M, Arsalan O, Alhaura L, Wijaya R, Syahrul Ramadhan AS, Yeremia C. Text steganography on digital video using discrete wavelet transform and cryptographic advanced encryption standard algorithm. In: Proceedings of the Sriwijaya International Conference on Information Technology and Its Applications (SICONIAN 2019); 2019 Nov 16; Palembang, Indonesia; 2020. p. 141–5. doi:10.2991/aisr.k.200424.021.
37. Goel A, Baliyan H, Tyagi S, Bansal N. End to end encryption of chat using advanced encryption standard-256. Int J Sci Res Arch. 2024;12(1):2018–25. doi:10.30574/ijsra.2024.12.1.0923.

38. Christopher C, Gunawan A, Prima S. Encrypted short message service design using combination of modified advanced encryption standard (AES) and vigenere cipher algorithm. EMACS J. 2022;4(2):73–7. doi:10.21512/emacsjournal.v4i2.8273.
39. Trung MM. Proposing an elliptic curve cryptosystem with the symmetric key for Vietnamese text encryption and decryption. Int J Adv Trends Comput Sci Eng. 2020;9(3):4158–62. doi:10.30534/ijatcse/2020/246932020.
40. Vincent OR, Okediran TM, Abayomi-Alli AA, Adeniran OJ. An identity-based elliptic curve cryptography for mobile payment security. SN Comput Sci. 2020;1(2):112. doi:10.1007/s42979-020-00122-1.
41. Patil PK, Bansode DR. Performance evaluation of hybrid cryptography algorithm for secure sharing of text & images. Int Res J Eng Technol. 2020;7(9):3773–8. doi:10.14445/22315381/ijett-v54p235.
42. Abu-Faraj MAM, Alqadi ZA. Using highly secure data encryption method for text file cryptography. Int J Comput Sci Netw Secur. 2021;21(12):53–60.
43. Sugirtham N, Jenny RS, Thiyaneswaran B, Kumarganesh S, Venkatesan C, Sagayam KM, et al. Modified playfair for text file encryption and meticulous decryption with arbitrary fillers by septenary quadrate pattern. Int J Networked Distrib Comput. 2024;12(1):108–18. doi:10.1007/s44227-023-00019-4.
44. Katz J, Lindell Y. Introduction to modern cryptography: principles and protocols. London, UK: Chapman and Hall/CRC; 2007.
45. Suneetha C, Surendra T, Neelima C. Implementation of double fold text encryption based on elliptic curve cryptography (ECC) with digital signature. Int J Recent Technol Eng. 2020;8(5):3840–6. doi:10.35940/ijrte.e6446.018520.
46. Sharma PL, Gupta S, Monga H, Nayyar A, Gupta K, Sharma AK. TEXCEL: text encryption with elliptic curve cryptography for enhanced security. Multimed Tools Appl. 2025;84(13):11503–31. doi:10.1007/s11042-024-19377-4.
47. Das P, Giri C. An efficient method for text encryption using elliptic curve cryptography. In: 2018 IEEE 8th International Advance Computing Conference (IACC); 2018 Dec 14–15; Greater Noida, India; 2019. p. 96–101. doi:10.1109/IADCC.2018.8692087.
48. Abu-Faraj M, Al-Hyari A, Aldebei K, Alqadi ZA, Al-Ahmad B. Rotation left digits to enhance the security level of message blocks cryptography. IEEE Access. 2022;10:69388–97. doi:10.1109/ACCESS.2022.3187317.
49. Alzyoud M, Saleh Alomar AM, Al-shanableh N, Al-Batah MS, Alqadi ZA, Al-Hawary SIS. The cryptography of secret messages using block rotation left operation. Appl Math. 2024;18(2):395–402. doi:10.18576/amis/180214.
50. Murari TV, KC R, ME R. Selective encryption of video frames using the one-time random key algorithm and permutation techniques for secure transmission over the content delivery network. Multimed Tools Appl. 2024;83(35):82303–42. doi:10.1007/s11042-024-18613-1.