Received: May 21, 2025. Revised: July 16, 2025.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Traceable Image Data Sharing through Blockchain and Reversible Watermarking

I Kadek Agus Ariesta Putra[1] Baskoro Adi Pratomo[1]* Hudan Studiawan[1]

[1]*Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia*
* Corresponding author's Email: baskoro@if.its.ac.id

Abstract: This paper proposes a traceable image data sharing system that integrates reversible watermarking and blockchain technology. A 1024-bit signature derived from image content and the applicant data, is embedded into the image as the watermark. We integrate blockchain technology to record copyright information and auxiliary watermarking variables of the shared data to provide immutable and transparent ownership records. To strengthen security, Arnold transform is applied to scramble the watermark and embedded in the discrete cosine transform (DCT) coefficients by taking into account the contrast sensitivity function to maintain imperceptibility. In the event of unauthorized sharing or misuse of image data, the embedded watermark can be extracted and traced, allowing us to identify and track responsibility for unauthorized image use. Experiments were conducted using 10 color images from diverse domains, including general images (USC-SIPI), medical datasets (CHASEDB1, ISIC, KVASIR), and biometric datasets (FERET). Experimental results show that the method achieves high imperceptibility with average peak signal-to-noise ratio (PSNR) values of 56.28 dB, and average structural similarity index (SSIM) scores close to 1.0 across test payloads ranging from 1 Kb to 16 Kb, robustness against common distortions and adversarial attacks with watermark normalized correlation (NC) scores above 0.92, and accurate traceability across 1,000 distributed copies with various distortions and adversarial attacks. Comparative evaluations confirm that the proposed method outperforms existing watermarking schemes in both robustness and traceability performance.

Keywords: Blockchain technology, Data traceability, Digital fingerprint, Image data sharing, Image watermarking, Security.

1. Introduction

The growth of the internet has enabled the widespread and instant distribution of images, both for personal and institutional purposes. In the medical field, for instance, hospitals often need to share patient scan results or medical images with other institutions for further diagnosis, collaborative research, or referrals between healthcare facilities.

However, the risk of security breaches arises when the data is distributed illegally by one of the recipient institutions. This situation becomes even more complex due to the difficulty of tracing the source of the leak, especially when the data is shared with many parties. Without a secure and structured data sharing mechanism, the personal information contained in the images could fall into the wrong hands, leading to potential misuse and violation of individual privacy.

Most current data sharing security systems focus only on access control, without any mechanism to trace the distribution trace when a leak occurs. Approaches such as encryption and authentication are indeed able to limit access to authorized parties but have not addressed the problem of tracing data after it is shared. This argument is reinforced by the review by Gupta et al. [1] showing that many image security technologies are still oriented towards protecting data confidentiality and integrity, but ignore the aspects of distribution tracing and post-access accountability. Similar findings were also found in the study of Chen and Huang [2], who developed fine-grained access controls on encryption systems, but did not include strategies to identify the source of data leakage after access rights were

 DOI: 10.22266/ijies2025.0930.27

granted. Wang [3] also showed that the main focus of current systems is still on securing during the access and sharing process, without attention to post-distribution tracing. This focus on access control over distribution tracking indicates a gap in the design of current image security systems, especially in the context of data sharing involving multiple parties.

In a data-sharing environment involving multiple parties, the ability to trace data distribution in detail is a key requirement. When an image is shared with various institutions or individuals, it increases the risk of the data being distributed illegally. Therefore, a security system that merely restricts access to data is not sufficient; a mechanism is needed that enables tracking of who receives the data, when, and in what context it was shared. Technical challenges arise because each copy of an image typically has an identical appearance. Therefore, an approach is required that allows tracking of distribution without altering the visual content of the image itself.

Watermarking is an ideal solution for embedding identity information into digital images without compromising the visual aspects of the image. Digital image watermarking is a technique that uses digital images as a carrier to embed additional data, either in the form of visible watermarks or invisible watermarks. In general, watermarking systems consist of two main stages, namely the embedding process and the extraction process. Based on their resistance to interference or manipulation, watermarks are divided into two categories: robust watermarks, which remain recognizable even if the image is compressed, cropped, or otherwise manipulated; and fragile watermarks, which are easily damaged and used to detect image integrity. In terms of where the data is inserted, watermarking techniques can be applied in the spatial domain, by inserting data directly into pixel values, or in the frequency domain, such as Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT), which offer higher resistance to compression and other visual attacks. Additionally, watermarking can be classified based on its reversibility. Irreversible watermarking does not allow the original image to be fully restored after embedding, while reversible watermarking enables the watermark information to be completely removed and the original image restored to its original form [5].

However, embedding a watermark alone is still not sufficient without a system capable of openly recording and verifying the distribution process [4]. Therefore, to support effective data traceability, an additional solution is required to ensure transparency and accountability across the entire digital data-sharing ecosystem.

The emergence of Bitcoin in 2008 marked the beginning of the development of blockchain technology as a decentralized digital infrastructure [5]. Although initially designed to support cryptocurrency systems, blockchain's main features such as transparency, immutability, and consensus mechanisms have attracted researchers to be applied to various applications outside the financial realm. In the context of digital data distribution, blockchain allows the recording of transactions such as the time of distribution, the identities of the sender and recipient, and other metadata into blocks that are chronologically linked and cannot be changed without network consensus. By storing every distribution activity into a distributed digital ledger, this system allows independent verification by all parties involved, making it a very relevant advantage in data sharing systems that require high accountability.

To embed copyright information and prevent intellectual property violations, various approaches have been developed in image watermarking technologies. Faheem et al. [6] combined the Least Significant Bit technique with image gradient analysis to determine optimal embedding locations, further enhanced by chaotic map-based encryption to improve security. Mohammed et al. [7] proposed a blind watermarking method that integrates DCT-DWT transformation and adaptive color channel selection to maintain robustness and imperceptibility. Abadi and Moallem [8] introduced a hybrid approach based on DWT and DCT, incorporating a three-stage voting system effective against noise-based distortions. Dong et al. [9] proposed a hybrid domain color image watermarking scheme that combines DWT, DCT, and Singular Value Decomposition (SVD), enhanced by watermark encryption using the Lorenz hyperchaotic map. Zhou et al. [10] proposed a robust image watermarking algorithm that embeds grayscale watermark pixel values directly into the DCT domain of a color host image, combined with block selection and geometric correction.

In the domain of reversible watermarking, Li et al. [11] refined generalized histogram shifting techniques to improve resistance against misalignment attacks. Fan et al. [12] applied modulo operations and prediction-error expansion on interpolated images to enhance embedding capacity while preserving visual quality. Similarly, Tanwar and Panda [13] developed a hybrid method combining histogram shifting and prediction error expansion with a local variance-based embedding strategy to reduce distortion.

Watermarking plays a central role when integrated with Digital Rights Management (DRM)

to verify ownership and restrict data access [14]. Traditional DRM systems are often constrained by centralized storage, raising concerns about vulnerabilities such as data tampering and copyright manipulation. In response, blockchain technology has gained traction in DRM applications to address these limitations. For instance, the DRPChain system [15] embeds copyright information into digital images in the form of QR codes as watermarks, with watermarked images stored off-chain using InterPlanetary File System (IPFS). Copyright metadata, perceptual hashes, IPFS addresses, and digital signatures are registered in each blockchain transaction. Additionally, redactable blockchain approaches have introduced mechanisms for illegal content removal through chameleon hashing, while preserving originality verification via perceptual hashing and implementing a reputation-based incentive system [16].

However, DRM alone is insufficient to address the broader challenges of data sharing. DRM systems primarily focus on copyright protection and distribution control. For example, in typical DRM implementations, a digital image may be watermarked with owner information solely for proving ownership in the event of misuse. In contrast, secure data-sharing scenarios require personalized watermarks to be embedded each time the data is distributed to a recipient, allowing traceability in the event of unauthorized redistribution.

Several studies have addressed user traceability in the context of unauthorized data distribution. The BE-TRDSS scheme [17], for instance, utilizes Ciphertext-Policy Attribute-Based Encryption with hidden access policies and stores ciphertext indexes and revocation lists on the blockchain. This enables the revocation of malicious users through identity tracking embedded in private keys. Wang and Guan [18] proposed a system that stores encrypted data on IPFS, secures the encryption hash using Elliptic Curve Cryptography (ECC), and records data-sharing activities on a smart contract–driven visual log. This enables real-time traceability for both data owners and recipients. In the medical domain, MRDACE [19] logs medical access metadata on a permissioned blockchain using a Proof-of-Authority mechanism. It supports tamper-resistant activity logging and automated decision-making through incentive functions that evaluate data requester privilege scores. The system also allows anonymous access for research institutions while preserving patient ownership controls. Meanwhile, Lai et al. [20] introduced a certificateless traceable ring signature scheme based on distributed key generation to ensure user traceability while maintaining privacy in Electronic Health Record (EHR) sharing environments. Their system employs Self-Controlling Objects (SCOs) for decryption management, stores data on IPFS, and leverages blockchain as an audit trail and authorization layer for proxy-based data sharing nodes.

In this study, we propose an image data sharing approach that combines reversible watermarking techniques and blockchain technology to build an image data sharing system that focuses on data traceability. The watermarking technique is used to embed a fingerprint created from the combination of the shared data with the requester's information. Meanwhile, blockchain is utilized as a decentralized metadata storage infrastructure. Information related to the data distribution process, such as the data owner, data requester, timestamp, and watermarking technique variables are stored as a transaction on the blockchain.

Unlike traditional DRM systems that mainly focus on ownership verification and unauthorized access prevention, our method provides data traceability by embedding user-specific fingerprints for each image distribution. In contrast to robust watermarking schemes, which generally do not allow the recovery of the original image because they happen to be irreversible, our approach utilizes reversible watermarking to ensure that the original image can be fully recovered, making it suitable for medical images because data integrity matters. Previous blockchain-based DRM solutions typically record only copyright metadata do not perform any verification or testing regarding data traceability. Our proposed method fills this gap through combining reversible watermarking with blockchain-based distribution metadata storage. This integration enables data tracking and identification of misuse origin even after multiple redistributions.

The remainder of this paper is organized as follows. Section 2 presents the preliminaries. Section 3 outlines the proposed method. Section 4 discusses the results and analysis. Section 5 describes the implementation of the proposed system, and Section 6 concludes the paper.

2. Preliminaries

2.1 Discrete cosine transform

Discrete Cosine Transform (DCT) is a widely used technique for converting spatial domain image data into the frequency domain. Originally proposed by Ahmed et al. [21], DCT is a special case of the Fourier transform and is renowned for its energy compaction and computational efficiency. For a

grayscale image $f(x,y)$ of size $N \times N$, the forward two-dimensional DCT is defined in Eq. (3), where the resulting DCT coefficients are denoted by $F(u,v)$ as computed in Eq. (1), $u,v = 0,1,\dots,N-1$, and the scaling factors are defined in Eq. (2). The inverse transformation, which reconstructs the image from the frequency coefficients, is defined in Eq. (3).

$$a(u)a(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)cos\left[\frac{(2x+1)u\pi}{2N}\right]cos\left[\frac{(2y+1)v\pi}{2N}\right] \quad (1)$$

$$a(u) = \begin{cases} \frac{1}{\sqrt{N}}, & u = 0 \\ \sqrt{\frac{2}{N}}, & u \neq 0 \end{cases}, a(v) = \begin{cases} \frac{1}{\sqrt{N}}, & v = 0 \\ \sqrt{\frac{2}{N}}, & v \neq 0 \end{cases} \quad (2)$$

$$\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} a(u)a(v)F(u,v)cos\left[\frac{(2x+1)u\pi}{2N}\right]cos\left[\frac{(2y+1)v\pi}{2N}\right] \quad (3)$$

2.2 Human visual system and RGB- YCbCr color space

The Human Visual System (HVS) exhibits non-uniform sensitivity to spatial frequencies, where the eye responds most strongly to frequencies in the mid-range and less effectively to very low or very high ones. This behaviour can be modelled mathematically by the Contrast Sensitivity Function (CSF), which quantifies the visibility threshold of contrast at a given spatial frequency. One of the most referenced models is proposed by Mannos and Sakrison [22], as shown in Eq. (4), where human perception is shown to be dominated by luminance sensitivity over chrominance.

In the frequency domain when applying the DCT, the CSF can be used to construct a perceptual weight matrix that regulates the insertion strength based on the sensitivity of each frequency component. Each weight is calculated by considering the CSF at the spatial frequency $f(u,v)$, obtained from the DCT index position (u,v), as defined in Eq. (5). This perception-aware weighting enables adaptive and efficient watermarking by emphasizing components with lower visual significance.

YCbCr is a widely used color space in digital photography and video processing pipelines. It separates an image into one luminance component (Y) and two chrominance components (Cb and Cr), which represent the blue-difference and red-difference color information, respectively [23]. The transformation from RGB to YCbCr is given in Eq. (6), while the inverse transformation from YCbCr back to RGB is described in Eq. (7).

$$CSF(f) = 2.6 \cdot (0.0192 + 0.114f) \cdot e^{-(0.114f)^{1.1}} \quad (4)$$

$$\gamma(u,v) = \frac{CSF(f(u,v))}{max(CSF)}, f(u,v) = \frac{1}{N} \cdot \sqrt{u^2+v^2} \quad (5)$$

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0 \\ 0.5 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 0.29890 & 0.58660 & 0.11450 \\ -0.16874 & -0.33126 & 0.50000 \\ 0.50000 & -0.41869 & -0.8131 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.40200 \\ 1 & -0.34414 & -0.71414 \\ 1 & 1.7720 & 0 \end{bmatrix} \begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} - \begin{bmatrix} 0 \\ 0.5 \\ 0.5 \end{bmatrix} \quad (7)$$

2.3 Logistic map

The logistic map is a simple yet powerful chaotic system widely used in secure image processing due to its pseudo-random behavior and high sensitivity to initial conditions. It is mathematically described in Eq. (8) where $x_n \in (0,1)$ and the control parameter r is typically chosen within $3.57 < r \leq 4$ to ensure fully chaotic behavior. By starting from an initial value x_0 as a secret key, the logistic map produces a deterministic but unpredictable sequence of real numbers within the unit interval.

$$x_{n+1} = r\,x_n(1-x_n) \quad (8)$$

In image watermarking, these chaotic sequences can be used to generate secure block orders. For example, after generating a logistic sequence of length equal to the number of image blocks, sorting the sequence and assigning ranks provides a unique pseudo-random permutation of block indices. This randomized ordering greatly increases security, since only users with knowledge of x_0 and r can reproduce the correct sequence to embed or extract the watermark.

2.4 Two-dimensional Arnold transform

The two-dimensional Arnold transform is a widely adopted technique for image scrambling. It operates on a matrix of size $N \times N$, and its forward transformation is defined in Eq. (9), where (x_1, y_1) denotes the coordinates of a pixel in the original image, and (x_2, y_2) represents the coordinates after transformation. The parameter N indicates the dimension of the square image matrix. To reverse the transformation and restore the original image, the inverse Arnold transform is applied as defined in Eq. (10).

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \bmod N \quad (9)$$

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} \bmod N \qquad (10)$$

2.5 InterPlanetary file system

The InterPlanetary File System [24] is a decentralized peer-to-peer protocol which designed for the purpose of retrieving file or data as well as storing them depending on their respective content, instead of location. IPFS uses content addressing through cryptographic content identifiers (CID) in order to retrieve data by using its unique hash, unlike customary web systems that rely on location-based addressing (e.g., URLs). Merkle Directed Acyclic Graphs (Merkle DAGs) are generally used to organize content within IPFS and each node (file or directory) is distinctly identified by its CID. This structure has immutability, deduplication, as well as tamper resistance because any node modification will definitely change its hash and propagate to all connected ancestors forming an entirely new DAG. CID linking by way of the Merkle DAG allows for representation of the complete file system. This system can then be verified in an efficient manner. A Distributed Hash Table (DHT) is used by IPFS with content discovery for discovering all peers and also data across the network without needing any central server.

2.6 Elliptic curve digital signature algorithm

Elliptic Curve Digital Signature Algorithm (ECDSA) is a lightweight cryptographic scheme based on the mathematics of elliptic curves. It is widely used to sign messages in a secure and concise manner [25]. The scheme starts by generating a key pair (d, Q), where d is the private signing key and $Q = dG$ is the corresponding public key derived using scalar multiplication on the elliptic curve.

To sign a message m, the signer chooses a random nonce k, such that $1 \leq k \leq n-1$. Then, calculate the elliptic curve points $(x_1, y_1) = kG$, and set $r = x_1 \bmod n$. The final signature is a pair (r, s), both of which must be nonzero. The nonce k must remain secret and unique for each message.

2.7 Blockchain technology

Blockchain is a decentralized and distributed ledger system that allows data to be recorded securely, transparently, and immutably across a network of computers. Originally introduced as the foundational technology behind Bitcoin [5], blockchain has since evolved into a general-purpose infrastructure applicable to a wide range of domains, including finance, supply chain, healthcare, and digital data management. At its core, a blockchain consists of a series of data blocks that are chronologically linked using cryptographic hashes. Each block typically contains a set of transactions, a timestamp, a reference to the previous block (in the form of a hash), and a nonce used in the consensus mechanism. This structure ensures that any alteration to a block's content will invalidate the subsequent chain, providing strong tamper-evidence.

To achieve distributed agreement on the state of the ledger, blockchain networks implement consensus algorithms such as Proof of Work (PoW), Proof of Stake (PoS), or other variants. These mechanisms enable all participating nodes to agree on a single version of the truth without relying on a centralized authority. One of the defining features of blockchain is its immutability. Once data has been recorded and confirmed by the network, it cannot be modified retroactively without consensus from the majority of the network.

2.8 Comparative watermarking methods

To evaluate the effectiveness of the proposed method, this study includes two existing watermarking schemes as comparative baselines: the method by Dong et al. [9] and the method by Zhou et al. [10]. The details of each method are described in the following subsections.

2.8.1. Overview of Dong's algorithm

Dong et al. [9] proposed a hybrid domain color image watermarking scheme that combines DWT, DCT, and Singular Value Decomposition (SVD), enhanced by watermark encryption using the Lorenz hyperchaotic map. DWT is a multi-resolution transform that decomposes an image into sub-bands LL (Low-Low), LH (Low-High), HL (High-Low), and HH (High-High), where the LL sub-band captures the main low-frequency information with strong stability, making it suitable for watermark embedding. SVD is a matrix factorization technique that decomposes an image matrix A into three matrices U, Σ, and V^T, expressed as $A = U\Sigma V^T$, where U and V are orthogonal matrices and Σ is a diagonal matrix containing the singular values σ_i. These singular values represent the intrinsic energy and structural features of the image and are relatively robust to compression, noise, or geometric changes, making them effective for watermark embedding. In this method, the embedded watermark has a fixed size of exactly one-fourth the dimensions of the cover image (i.e., if the cover is $M \times N$, the watermark is $\frac{M}{2} \times \frac{N}{2}$.)

The key steps of the data embedding and extraction can be summarized as follows:

2.8.1.1. Watermark embedding steps:

1. Read the grayscale watermark and encrypt it with a Lorenz hyperchaotic map
2. Perform SVD on the encrypted watermark to get U_w, S_w, V_w
3. Read the cover image, convert to YCbCr color space, and select the Y channel
4. Compute the embedding factor α adaptively using Bhattacharyya distance between the watermark and the cover image
5. Perform one-level DWT on the Y channel to obtain four sub-bands: LL, LH, HL, and HH
6. Apply DCT on the LL sub-band (LLD)
7. Perform SVD on the transformed LLD to get U_H, S_H, V_H
8. Blend singular values: $S'_H = \alpha \times S_w + (1 - \alpha) \times S_H$
9. Reconstruct LLD' by inverse SVD: $LLD' = U_H \times S'_H \times V_H^T$
10. Apply inverse DCT on LLD' to reconstruct LL'
11. Rebuild the Y component with inverse DWT of LL', LH, HL, HH
12. Convert YCbCr back to RGB to obtain the final watermarked image

2.8.1.2. Watermark extraction

1. Read the watermarked image, convert to YCbCr, select the Y component
2. Apply one-level DWT to obtain $LL_{WM}, LH_{WM}, HL_{WM}, HH_{WM}$
3. Apply DCT to LL_{WM}
4. Perform SVD on DCT(LL_{WM}) to get U_{WM}, S_{WM}, V_{WM}
5. Recover the singular values (S'_W) of the encrypted watermark using Eq. (11), where S_{WM} is the singular value matrix extracted from the watermarked image, and S_H is the stored singular value matrix of the original host image.
6. Reconstruct encrypted watermark using inverse SVD: $W'_E = U_w \times S'_w \times V_w^T$
7. Decrypt using inverse Lorenz hyperchaotic mapping to get the final extracted watermark.

$$S'_w = \frac{S_{WM} - (1 - \alpha) \times S_H}{\alpha} \tag{11}$$

2.8.2. Overview of Zhou's algorithm

Zhou et al. [10] proposed a robust image watermarking algorithm that embeds grayscale watermark pixel values directly into the DCT domain of a color host image. The key steps of the watermark embedding and extraction can be summarized as follows:

2.8.2.1. Watermark embedding steps:

1. Scramble the grayscale watermark image using the Arnold transform with a secret key
2. Convert the host RGB image to the YCbCr color space and select the Cb channel
3. Apply one-level DWT on the Cb channel to decompose it into four sub-bands (LL, LH, HL, HH)
4. Select the LL sub-band for watermark embedding and divide it into non-overlapping 8×8 blocks
5. Calculate the block texture values using a DCT-based texture algorithm and sort the blocks by ascending texture (smoother blocks get priority). Store the block order as secret key Kb
6. For each selected block, subdivide it into four non-overlapping 4×4 sub-blocks
7. Perform 2D DCT on each 4×4 sub-block to obtain its frequency coefficients
8. Calculate the average direct current (DC) coefficient across the four sub-blocks and set this as the new DC coefficient reference
9. Embed the scrambled watermark pixel values as a ratio into two low-frequency alternating current (AC) coefficients of each DCT block, using the Eq. (12) where w is the water mark pixel, and k is the embedding strength factor
10. Apply inverse DCT to reconstruct the modified 4×4 sub-blocks, then combine them back to form the modified 8×8 block
11. After processing all selected blocks, perform inverse DWT on the modified LL together with unchanged LH, HL, HH to reconstruct the modified Cb component, then convert back from YCbCr to RGB to obtain the watermarked image

$$AC_w = \text{sign}(AC) \times \frac{\text{w} \times \text{k}}{DC + 1} \tag{12}$$

2.8.2.2. Watermark extraction steps:

1. Convert the watermarked RGB image to YCbCr color space and select the Cb channel
2. Apply one-level DWT to obtain LL, LH, HL, HH
3. Select the LL sub-band and divide it into 8×8 non-overlapping blocks
4. Use the stored block ordering key (Kb) to identify the watermarked blocks
5. For each watermarked block, subdivide into four 4×4 sub-blocks

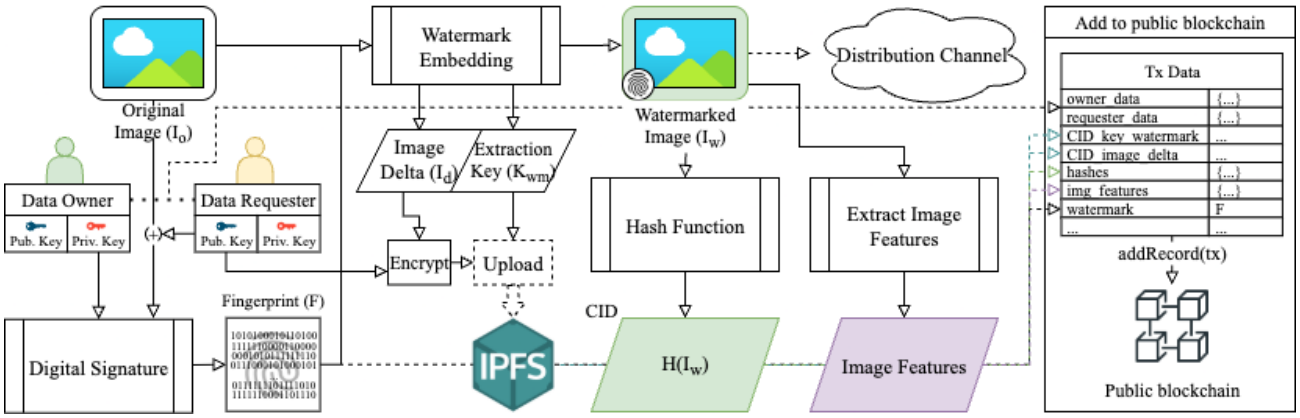


Figure. 1 General design of the proposed system

6. Perform 2D DCT on each sub-block to extract the modified frequency coefficients
7. Recover the embedded watermark pixel values from the two low-frequency AC coefficients, using Eq. (13)
8. Combine the recovered pixel values in the correct sequence to reconstruct the scrambled watermark
9. Apply the inverse Arnold transform with the secret key to recover the final extracted watermark image.

$$AC_w = \left| \frac{AC \times (DC + 1)}{k} \right| \quad (13)$$

3. Proposed method

An overview of the process of the proposed concept is shown in Fig. 1. The process requires the private key of the data owner and the public key of the data requester. The initial stage involves creating a digital signature. We utilize the ECDSA cryptographic algorithm based on the BRAINPOOLP512r1 curve [26], which produces a 1024-bit digital signature. This signature is generated by signing the concatenation of the image hash and the requester's public key. This fingerprint is embedded into the image as the primary watermark payload.

The watermark embedding process produces three outputs: the image delta which is the pixel difference value between the watermarked image with the original image, the extraction key to extract the watermark, and the watermarked image itself. In the context of medical images, reversibility is an important aspect because the original image contains critical information for diagnosis.

To achieve reversibility, image delta is used to recover the original image. The image delta is encrypted using the data requester's public key and stored in the IPFS as well as the extraction key, so that only the data requester can recover the original image, even though the image delta information is publicly accessible. Next, the hash value of the watermarked image is calculated, followed by the extraction of image features.

We utilize blockchain technology to store information related to the watermarking process and data distribution information. Information regarding the identity of the data owner, identity of the data requester, fingerprint value, CID of the extraction key and image delta in IPFS, hash value of the watermarked image, and image features are recorded as transactions on the public blockchain. The watermarked image can then be shared with the data requester.

3.1 Proposed watermarking schemes

3.1.1. Data embedding process

The watermark embedding process, as illustrated in Fig. 2, begins by dividing the cover image into blocks of size 8×8. Next, a block sequence is generated using a logistic map controlled by secret key parameters x_0, and r. After generating the block order, each block is converted to the YCbCr color space. The secret bits are embedded into the Y, Cr, Cb components of the image. Each DCT block has 8×8 size, containing 64 coefficients ordered containing 64 coefficients ordered using zigzag scanning. Each coefficient is denoted by $D(q)$, where $q \in 1,2,3, \ldots ,64$.

A scrambled watermark b of B bits is then produced using the Arnold transform, with the iteration count acting as the secret key parameter K_a. Each $n-th$ watermark bit (b_n) is embedded iteratively in R different locations of the DCT coefficients to improve robustness against attacks. The embedding locations are determined based on the index $q_n^{(k)}$, where q_s is the starting point, and R is the

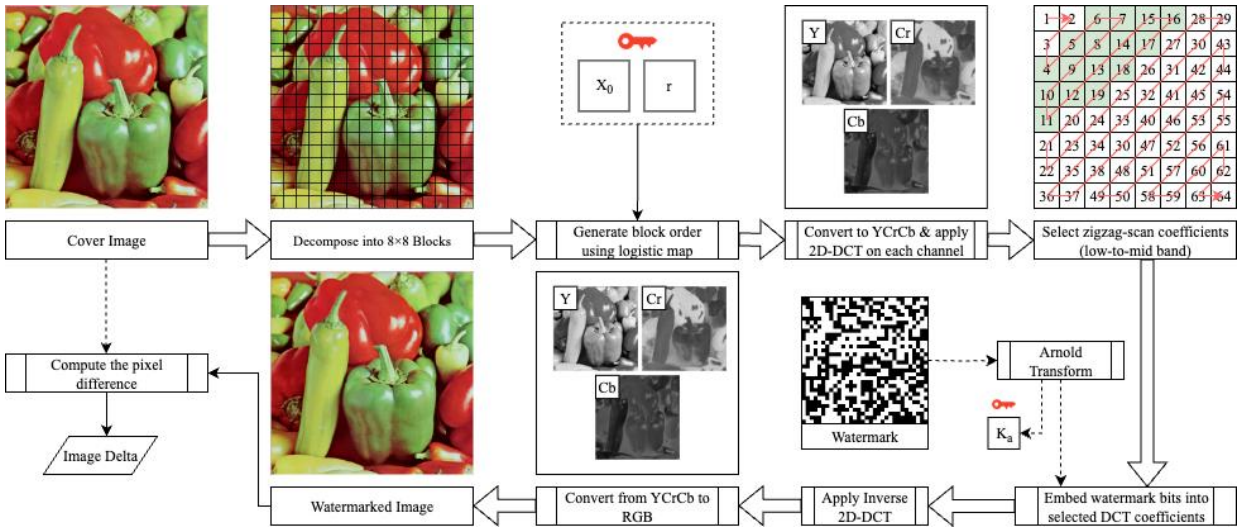


Figure. 2 The proposed watermark embedding scheme

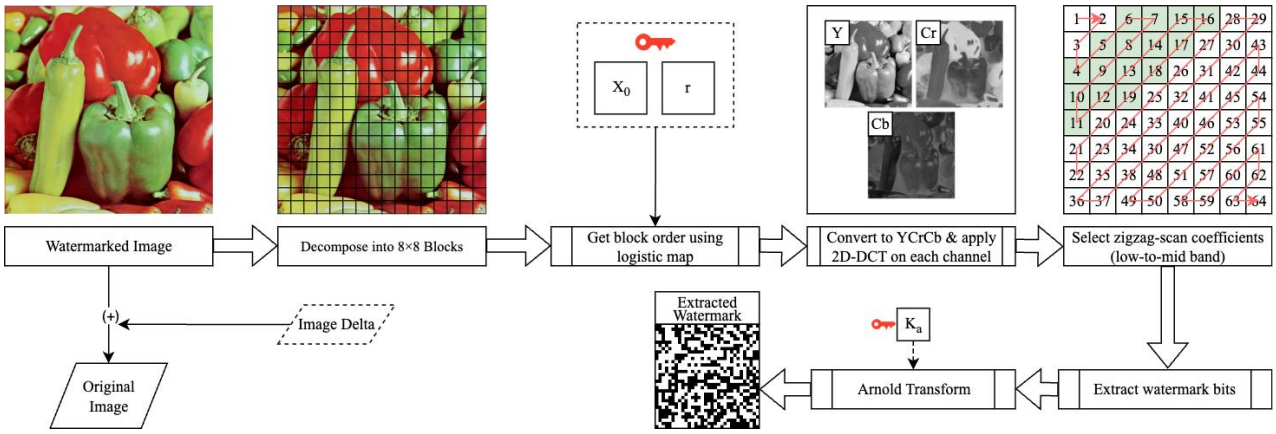


Figure. 3 The proposed watermark extraction scheme

embedding count per bit. The embedding position is defined in Eq. (14).

$$q_n^{(k)} = q_s + (k-1)B + (n-1), \quad for\ k = 1,2,\dots,R \tag{14}$$

Each watermark bit b_n is represented as a value w_n, where $w_n = +1$ if $b_n = 1$, and $w_n = -1$ if $b_n = 0$, as defined in Eq. (15).

$$w_n = \begin{cases} +1, & if\ b_n = 1 \\ -1, & if\ b_n = 0 \end{cases} \tag{15}$$

The DCT coefficient at the defined position is then modified using Eq. (16), where α (alpha) is the embedding strength, $\gamma\left(q_n^{(k)}\right)$ represents the perceptual weight corresponding to the frequency component located at position $q_n^{(k)}$.

$$D'\left(q_n^{(k)}\right) = D\left(q_n^{(k)}\right) + \alpha \cdot \gamma\left(q_n^{(k)}\right) \cdot w_n \tag{16}$$

The original coefficient value is stored as a key reference for future extraction. The variable $K_{p,q}$ stores the original DCT coefficient value before modification, where p denotes the index of the 8×8 block decomposed into DCT domain, and q denotes the zigzag index of the coefficient in block p as defined in Eq. (17).

$$K_{p,q} = D\left(q_n^{(k)}\right) \tag{17}$$

In this implementation, watermark bits are embedded into the intermediate-frequency coefficients of the 8×8 DCT block, following the zigzag scan order. Specifically, the embedding process starts from coefficient index $q_s = 3$ and continues up to index 19, which offer a good balance between image quality and robustness against image processing attacks.

3.1.2. Data extraction process

As illustrated in Fig. 3, the watermark extraction process begins by decomposing the watermarked

```
pragma solidity ^0.8.0;

contract PublicLog {
    event RecordAdded(
        address indexed sender, string data, uint timestamp
    );

    function addRecord(string memory data) public {
        emit RecordAdded(
            msg.sender, data, block.timestamp
        );
    }
}
```

Figure. 4 Solidity smart contract

image into blocks of size 8×8, with the block sequence determined using the extraction keys x_0 and r held by the data owner. Each block is then converted to the YCbCr color space, followed by applying the 2D-DCT on each channel. Watermark extraction is done by comparing the watermarked DCT coefficients with the original DCT coefficients stored as references (e.g., in the blockchain). The difference between these values is calculated and summed to estimate the hidden watermark in the image. The cumulative value for each bit n is computed as defined in Eq. (18).

$$\tilde{S}_n = \sum_{k=1}^{R} \left(D' \left(q_n^{(k)} \right) - D \left(q_n^{(k)} \right) \right) \tag{18}$$

Once the cumulative value $\tilde{S}_n$ is obtained, the watermark bit is extracted by checking whether the value is greater than zero. The extracted watermark bit $\hat{b}_n$ is determined as defined in Eq. (19):

$$\hat{b}_n = \begin{cases} 1, & if\ \tilde{S}_n > 0 \\ 0, & if\ \tilde{S}_n \leq 0 \end{cases} \tag{19}$$

Unscrambling the watermark requires the Arnold transform key Ka. To restore the original image, the image delta values are added to the watermarked image to produce an estimate that closely approximates the original image.

3.2 Blockchain design

In this system, we leverage features that ensure data integrity and transparency, such as non-repudiation, traceability, and the distribution of data control, such as key reference for watermark data extraction. To store the necessary information, we utilize a public blockchain as the underlying database.

Table 1. Structure of the encoded payload format

Field No.	Description	Example
1	Owner name	Alice
2	CID of owner public key	QmRv...ecrR
3	Requester name	Bob
4	CID of requester public key	QmQ2...VjjW
5	Watermark fingerprint	0x31c8fb19...
6	CID of watermark extraction key	QmR2...reeW
7	CID of image delta	QmNg...BqSm
8	Image height	512
9	Image width	512
10	PDQ hash of the image	a9c3e1...
11	Hash of the original image	f13bcd...
12	Hash of the watermarked image	cdd492...

This public blockchain allows anyone to access the stored data while ensuring that the data remains immutable and cannot be repudiated by any party. The data stored in the blockchain includes:

- Information of the data owner
- Information of the data demander
- IPFS CID of the watermarking extraction key
- IPFS CID of the image delta
- Hash values of original and watermarked image
- Feature values of the image (e.g., height, width, PDQ hash, etc.)
- The embedded watermark value (fingerprint)

Each of these elements contributes to traceability and the authenticity of information. The data recorded on the blockchain does not contain confidential content, but rather serves to support authentication and verification processes. For example, the extraction key in the watermarking process does not pose a disclosure risk, as it is solely used to verify the presence of a watermark rather than to conceal sensitive information. Furthermore, critical extraction keys such as those used in the logistic map are not stored, they remain known only to the data owner.

To optimize performance and reduce on-chain storage costs, we deployed a lightweight smart contract on the Polygon Mainnet. The contract refered as *PublicLog*, is shown in Fig. 4, is implemented in Solidity and uses an event-based logging mechanism.

To reduce on-chain gas consumption, we used a compact string format to encode all distribution metadata into a single payload string. Instead of using verbose key-value pairs or JSON structures, the payload is composed of sequential fields separated by pipe symbols "|". Each field holds a specific piece of metadata. This format is designed for minimal

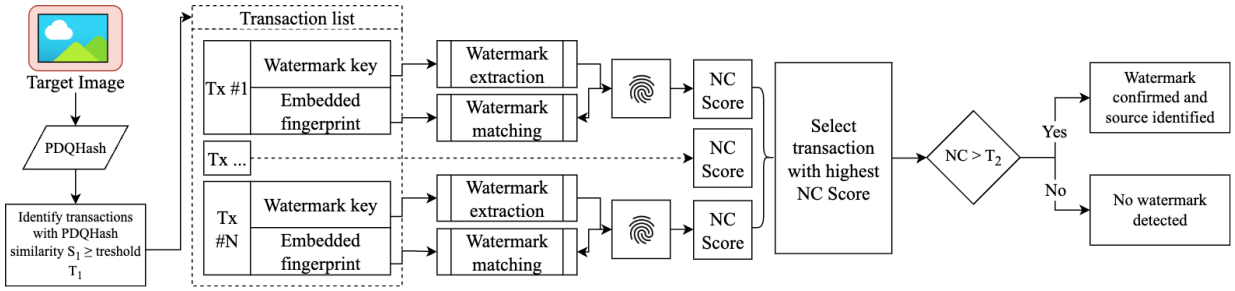


Figure. 5 Data identification process from blockchain records

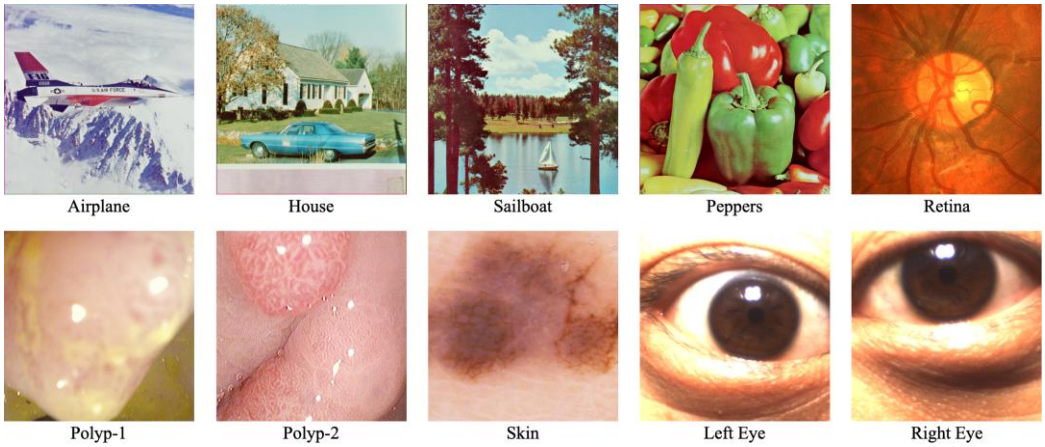


Figure. 6 Image set for experiment

storage overhead. The structure of the encoded payload is shown in Table 1. To further reduce on-chain size and optimize cost efficiency, the actual public keys of both the data owner and the requester are stored off-chain in the IPFS. Only the content identifiers of these public key files are recorded on the blockchain. This approach avoids the high storage cost associated with recording large key strings on-chain.

3.3 Data identification process

As illustrated in Fig. 5, the process begins by computing the perceptual hash (PDQHash) of the target image to extract its perceptual features. These features are then compared with hashes stored in blockchain transactions. Transactions with a PDQHash similarity score (S^1) greater than or equal to a predefined threshold T^1 are shortlisted as potential matches. For each matched transaction, the system performs watermark extraction using the associated watermark key and also retrieves the expected fingerprint. This fingerprint is then compared against the extracted one through a watermark matching process to calculate a Normalized Correlation (NC) score. The system then selects the transaction with the highest NC score. If the highest score exceeds a second threshold T_2, the watermark is considered valid and the associated transaction is confirmed as the origin of the image. Otherwise, the system concludes that no watermark is detected or the source cannot be reliably identified.

4. Experiment results and discussions

4.1 Experiment setup

The complete set of test images used in this experiment is presented in Fig. 6. The experiments were conducted using 10 color images originating from diverse application domains, including general images as well as medical images. All images had a resolution of 512×512 pixels. Four of these images, Airplane, House, Peppers, and Sailboat were obtained from the USC-SIPI Image Database [27]. In addition, this study incorporated medical images, such as the Retina image from the CHASEDB1 dataset [28]. Two colon polyp images (Polyp-1 and Polyp-2) were selected from the KSAVIR dataset [29], which provides endoscopic data for gastrointestinal lesion detection. The Skin image was collected from the ISIC dataset [30]. Finally, two eye images, namely “Left Eye” and “Right Eye”, were obtained from the FERET dataset [31], which is frequently used in biometric research involving iris and facial recognition.

To evaluate the effectiveness of the proposed method, comparative experiments were conducted by

embedding the same watermark payload into each image using three different approaches: (1) the proposed method, (2) the method by Dong et al. [9], and (3) the method by Zhou et al. [10], as previously described in Section 2.

4.1.1. Experimental environment

The experiments were conducted on a MacBook Air device equipped with an Apple M1 chip, 8 GB of RAM, and a 256 GB SSD. The operating system used was macOS, and the experiments were implemented using Python version 3.11.10.

4.1.2. Imperceptibility evaluation setup

This experiment investigates the effect of varying alpha values on image quality by measuring the Peak Signal-to-Noise Ratio (PSNR) across all color channels. PSNR is computed based on the mean squared error (MSE) between the original image $I(i,j)$ and the watermarked image $K(i,j)$, as defined in Eq. (20). Here, m and n represent the image dimensions. Higher PSNR values indicate lower distortion and thus better imperceptibility.

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{255^2}{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}[I(i,j) - K(i,j)]^2}\right) \quad (20)$$

The next experiment evaluates the impact of different payload sizes (1 Kb, 2 Kb, 4 Kb, 8 Kb, and 16 Kb) on image quality using both PSNR and the Structural Similarity Index Measure (SSIM). SSIM evaluates perceptual similarity based on luminance μ, contrast σ, and structural correlation σ_{xy}, as described in Eq. (21). Constants C_1 and C_2 are used to stabilize the calculation against division by zero. SSIM values close to 1 imply near-identical structural content between the original and watermarked images.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (21)$$

4.1.3. Robustness evaluation setup

Robustness is assessed using normalized corelation (NC) between the original watermark $W(i)$ and the extracted watermark $\widehat{W}(i)$, as shown in Eq. (22). The value N represents the total number of bits in the watermark. An NC value close to 1 indicates strong similarity and effective recovery.

$$\text{NC} = \frac{\sum_{i=1}^{N} W(i) \cdot \widehat{W}(i)}{\sqrt{\sum_{i=1}^{N} W(i)^2} \cdot \sqrt{\sum_{i=1}^{N} \widehat{W}(i)^2}} \quad (22)$$

This experiment examines the robustness of the proposed method under various image distortion and adversarial attacks. The tested scenarios include: no attack, JPEG2000 compression (Quality Factor = 30), salt-and-pepper noise (density = 0.05), Gaussian noise (variance = 0.01), Gaussian blur (3×3 kernel size, $\sigma = 1$), sharpening ($\alpha = 1.5$), median filtering (3×3), contrast adjustment ($\alpha = 1.5, \beta = 0$), resizing (256×256 pixels), black-box insertion (75×75 pixels).

In addition, a copy attack scenario is included, which is an atempt to remove or weaken the embedded watermark by averaging multiple watermarked images. This approach assumes that an attacker has access to several watermarked versions of the same image, and generates a composite by averaging the pixel values across them. The resulting image is expected to dilute the watermark signal, potentially making detection more difficult. In this experiment, the copy attack is configured using 10 reference images in addition to the target.

Furthermore, the robustness evaluation includes three adversarial attacks: Fast Gradient Sign Method ($\epsilon = 0.1$), Basic Iterative Method ($\epsilon = 0.5, \alpha = 0.5$), and Projected Gradient Descent ($\epsilon = 0.4, \alpha = 0.7$). Further details regarding the design and implementation of the adversarial attack scenarios are described in the following subsection.

4.1.3.1. Adversarial attack setup

Although the proposed watermarking scheme does not employ any machine learning models for embedding or extraction, this experiment includes an adversarial attack evaluation to test the resilience of the watermark against structured pixel perturbations. Adversarial attacks are traditionally designed to mislead deep learning classifiers by applying carefully crafted, imperceptible perturbations. To simulate this worst-case scenario, we employed a pre-trained MobileNetV2 classifier as a surrogate model. Adversarial perturbations were then generated against this model using three standard techniques Fast Gradient Sign Method, Basic Iterative Method, and Projected Gradient Descent:

a) Fast Gradient Sign Method

Fast Gradient Sign Method (FGSM) is a white-box adversarial attack technique that perturbs an input image by leveraging the gradient of the loss function with respect to the input. Specifically, it adds

a small perturbation in the direction of the sign of this gradient, scaled by a parameter ε, with the goal of maximizing the model's loss while minimally altering the image perceptually. Mathematically, the FGSM perturbation is defined in Eq. (23), where x is the original input image, J is the loss function, θ the model parameters, and ϵ the perturbation magnitude.

$$\mathrm{x_{adv}} = \mathrm{x} + \epsilon \times \mathrm{sign}\big(\nabla_{\mathrm{x}} \mathrm{J}(\theta, \mathrm{x}, \mathrm{y})\big) \tag{23}$$

For this experiment, we consider the $\epsilon = 0.01$, representing moderate level of adversarial perturbation.

b) Basic Iterative Method

In addition to FGSM, the Basic Iterative Method (BIM) was also applied to evaluate the robustness of the proposed watermarking approach under iterative adversarial perturbations. BIM is an extension of FGSM, which performs multiple small-step gradient-based updates instead of a single step. By repeatedly applying FGSM-like perturbations with a small step size and projecting the adversarial image back to a valid ε-neighborhood of the original image after each iteration, BIM can generate stronger adversarial examples with higher success rates of fooling a target model. Mathematically, BIM is described as an iterative process as defined in Eq. (24), where α is the step size, ϵ is the maximum perturbation budget, and $Clip_{x,\epsilon}$ ensures that the adversarial sample stays within the allowed range around the original input. As in the FGSM setup, a pretrained MobileNetV2 network was used as a surrogate to estimate the gradient and simulate a deep learning–based adversarial scenario.

$$x^{(i+1)} = \mathrm{Clip}_{x,\epsilon}\left\{x^{(i)} + \alpha \times \mathrm{sign}\left(\nabla_x J\left(\theta, x^{(i)}, y\right)\right)\right\} \tag{24}$$

For this experiment, BIM was configured with $\epsilon = 0.5$ and $\alpha = 0.5$, over 10 iterations, to provide a moderate but realistic adversarial perturbation scenario that can challenge the watermark extraction process.

c) Projected Gradient Descent

In addition to FGSM and BIM, the Projected Gradient Descent (PGD) method was also applied to further evaluate the adversarial robustness of the proposed watermarking scheme. PGD is similar to BIM in that it performs multiple small-step gradient-based updates according to the same iterative rule described in Eq. (24), but differs by introducing a random initialization within the ϵ-ball around the input image. This random start allows PGD to explore a broader region of the input space and avoid relying solely on a single starting point. After each update, the perturbation is projected back to stay within the allowed ϵ-neighborhood. As with the other attacks, a pretrained MobileNetV2 network was used as a surrogate model to compute the required gradients and generate the perturbations. In this experiment, PGD was configured with $\epsilon = 0.4$, $\alpha = 0.7$, and 10 iterations, to simulate a moderate and realistic adversarial scenario for testing the watermark extraction robustness.

4.1.4. Traceability evaluation setup

The next experiment aims to evaluate the traceability capability of the proposed watermarking scheme, while also comparing it with the methods of Dong et al. [9] and Zhou et al. [10] Each image was distributed to 100 recipients, with an identifying watermark embedded for each recipient, and the distribution records were stored on a public blockchain. After distribution, a total of 50 images were selected for source identification testing, consisting of 30 watermarked images recorded on the blockchain, 15 watermarked images not recorded on the blockchain, and 5 original images without watermarking.

All 50 test images were then subjected to the attack scenarios described in the previous experimental setup. Since each test image was tested under 10 different attack scenarios, this resulted in a total of 5,000 attacked image, which were then analyzed to identify their sources. Detection performance was evaluated using accuracy, precision, recall, and F1-score metrics, as defined in Eq. (25), (26), (27), and (28) respectively. The detection results were categorized as follows:

- True Positive (TP): a watermarked image recorded on the blockchain is correctly identified with its true source.
- True Negative (TN): an image not recorded on the blockchain (or an original image without watermarking) does not produce a detection result in the system.
- False Positive (FP): an image not recorded on the blockchain is incorrectly identified as if it were recorded.
- False Negative (FN): an image recorded on the blockchain is not correctly identified, or its detected source does not match the blockchain record.

The detection process is illustrated in Fig. 5, as described in the previous section. For this experiment,

we set the image similarity threshold to $T_1 = 80\%$ and the normalized correlation threshold to $T_2 = 0.7$.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (25)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (26)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (27)$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (28)$$

4.1.5. Blockchain benchmark setup

To benchmark the smart contract implementation in a real-world setting, we conducted a performance evaluation on the Polygon Mainnet. This simulation models a real-world image sharing scenario, in which each of the 10 test images was distributed to 100 recipients, resulting in 1,000 unique data-sharing transactions. Each transaction involves calling the *addRecord* function of the smart contract to record metadata such as data owner information (name and IPFS CID of the public key), data requester information (name and IPFS CID of the public key), embedded watermark fingerprint, watermark key IPFS CID, image features including perceptual hash, and the watermarked and original image hash, as detailed in the blockchain design. The test was designed to measure: (1) average gas cost per transaction, (2) estimated monetary cost based on Polygon Mainnet gas prices and token valuation, and (3) transaction throughput (transactions per second).

4.1.6. IPFS implementation and evaluation setup

IPFS was implemented on a virtual private server (VPS) with system specifications summarized in Table 2. To deploy IPFS, we installed go-ipfs version 0.21.0 on this server as the reference implementation of the IPFS protocol. A reverse proxy was configured using Nginx, which also handled HTTP Basic Authentication to restrict access during the upload process. The evaluation consisted of two main phases: uploading and retrieving watermark-related payloads (image delta files). In the upload phase, 1,000 files were added to IPFS, and their latency and upload availability were recorded. In the retrieval phase, each uploaded file was retrieved 10 times, resulting in a total of 10,000 retrieval operations. For each operation, latency and retrieval availability were measured to evaluate the performance of the IPFS node in a constrained server environment.

Table 2. Server specifications

Component	Specification
Provider	Vultr VPS (Microsoft Hyper-V virtualization)
CPU	1 vCPU Intel Core (Broadwell), 2.39 GHz
RAM	1 GB (951 MiB physical, 2.3 GB swap)
Storage	25 GB SSD
Operating System	Ubuntu 22.04.5 LTS, Linux kernel 5.15, x86_64
Location	Tokyo, Japan

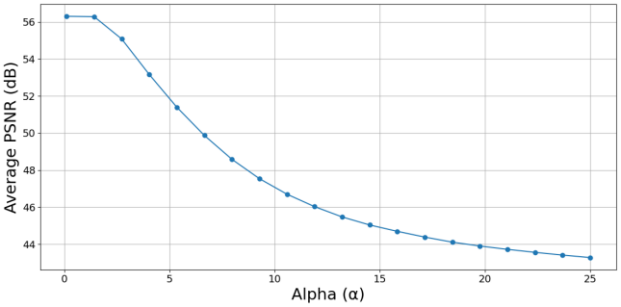


Figure. 7 Average PSNR by alpha value

4.2 Results and discussion

4.2.1. Effect of alpha values on PSNR across color channels

Fig. 7 shows the results of an experiment measuring the average PSNR values when a 10 Kb payload is embedded using different values of the embedding strength parameter α, ranging from 0.1 to 25. The embedding was performed using the proposed method with duplication factor $R = 1$. It can be observed that increasing the value of α leads to a gradual decline in PSNR, indicating reduced visual quality due to stronger watermark embedding. The highest PSNR is obtained when α is 0.1, reaching over 56 dB, while the lowest value is around 43.3 dB when α reaches 25. This trend highlights the trade-off between embedding strength and imperceptibility, where a stronger embedding (higher α values) improves watermark robustness at the cost of image quality.

4.2.2. Impact of payload size on PSNR and SSIM

Table 3 presents the results of the imperceptibility evaluation measured using PSNR on all test images. The embedded payloads range from 1 Kb to 16 Kb. For this experiment, the proposed method (Prop.) uses an embedding strength parameter of $\alpha = 0.1$. As presented in Table 3, the proposed method consistently produces higher PSNR values across all

Table 3. PSNR comparison by payload size

Image	1 Kb			2 Kb			4 Kb			8 Kb			16 Kb		
	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.
Airplane	38.14	47.33	**61.90**	38.11	45.17	**58.97**	38.00	42.99	**56.00**	37.98	40.43	**52.98**	38.03	37.45	**49.97**
House	38.68	50.88	**62.02**	38.64	47.16	**59.08**	38.53	44.85	**56.11**	38.50	41.68	**53.10**	38.56	38.81	**50.10**
Peppers	41.35	47.71	**62.08**	41.29	44.59	**59.18**	41.15	42.39	**56.21**	41.11	39.70	**53.21**	41.19	37.30	**50.19**
Sailboat	40.63	45.30	**62.11**	40.58	43.51	**59.17**	40.44	41.64	**56.15**	40.41	39.64	**53.12**	40.48	37.67	**50.11**
Skin	39.36	52.31	**62.45**	39.32	48.94	**59.56**	39.22	45.96	**56.59**	39.20	42.82	**53.62**	39.25	39.68	**50.61**
Polyp-1	38.70	52.79	**62.47**	38.66	50.22	**59.53**	38.56	47.26	**56.54**	38.56	43.92	**53.53**	38.61	40.65	**50.52**
Polyp-2	37.92	51.46	**62.49**	37.88	48.38	**59.56**	37.79	45.34	**56.57**	37.78	42.18	**53.53**	37.82	38.95	**50.53**
Retina	42.82	52.79	**62.45**	42.78	50.62	**59.56**	42.65	48.37	**56.58**	42.60	45.70	**53.57**	42.68	42.58	**50.56**
Left Eye	38.74	54.85	**62.14**	38.71	51.37	**59.13**	38.59	48.59	**56.13**	38.57	45.37	**53.11**	38.62	42.02	**50.12**
Right Eye	37.71	52.42	**62.18**	37.69	50.04	**59.21**	37.58	47.55	**56.24**	37.55	44.70	**53.19**	37.60	41.69	**50.18**
Average	39.41	50.78	**62.23**	39.37	48.00	**59.30**	39.25	45.49	**56.31**	39.23	42.61	**53.30**	39.28	39.68	**50.29**

Table 4. SSIM comparison by payload size

Image	1 Kb			2 Kb			4 Kb			8 Kb			16 Kb		
	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.
Airplane	0.997	0.998	**1.000**	0.997	0.997	**0.999**	0.997	0.996	**0.999**	0.997	0.992	**0.998**	0.997	0.983	**0.996**
House	0.997	0.999	**1.000**	0.997	0.999	**1.000**	0.997	0.998	**0.999**	0.997	0.996	**0.998**	0.997	0.991	**0.997**
Peppers	0.996	0.998	**1.000**	0.996	0.996	**1.000**	0.996	0.993	**0.999**	0.996	0.988	**0.999**	0.996	0.976	**0.997**
Sailboat	0.998	0.998	**1.000**	0.998	0.998	**1.000**	0.998	0.996	**0.999**	0.998	0.993	**0.999**	0.998	0.988	**0.997**
Skin	0.988	0.998	**1.000**	0.988	0.995	**0.999**	0.988	0.990	**0.999**	0.988	0.978	**0.998**	0.988	0.955	**0.995**
Polyp-1	0.996	0.999	**1.000**	0.996	0.998	**1.000**	0.996	0.996	**0.999**	0.996	0.991	**0.999**	0.996	0.979	**0.998**
Polyp-2	0.988	0.998	**1.000**	0.988	0.995	**0.999**	0.988	0.990	**0.999**	0.988	0.977	**0.998**	0.988	0.951	**0.996**
Retina	0.989	0.999	**1.000**	0.989	0.997	**0.999**	0.989	0.996	**0.999**	0.989	0.992	**0.998**	0.989	0.981	**0.996**
Left Eye	0.989	0.999	**1.000**	0.989	0.997	**0.999**	0.989	0.995	**0.999**	0.989	0.990	**0.997**	0.989	0.980	**0.995**
Right Eye	0.992	0.999	**1.000**	0.992	0.997	**0.999**	0.992	0.996	**0.999**	0.992	0.992	**0.998**	0.992	0.983	**0.996**
Average	0.993	0.999	**1.000**	0.993	0.997	**0.999**	0.993	0.995	**0.999**	0.993	0.989	**0.998**	0.993	0.977	**0.996**

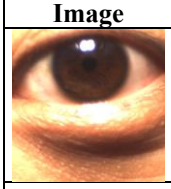
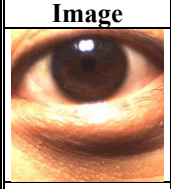
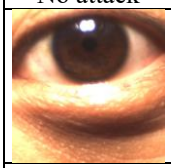
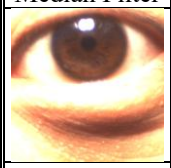
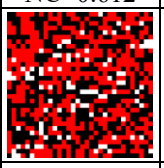
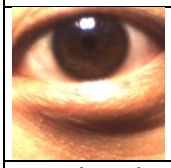
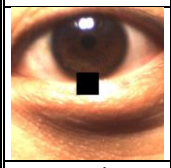
payload sizes and images. The PSNR values for the proposed method generally range from average 62.23 dB at 1 Kb to average 50.29 dB at 16 Kb, indicating a gradual decline in image fidelity as the payload size increases, which is expected in watermarking systems. The method by Zhou et al. [10] shows a more noticeable decrease in PSNR as the payload increases, with values in several cases falling below 40 dB at the highest payload size. This indicates that the visual impact of watermarking becomes more significant under heavier embedding in their approach. In contrast, the method by Dong et al. [9] yields relatively stable PSNR values across all payload sizes. This is attributed to the properties of Dong's method, which requires the watermark length to be fixed at one-fourth of the image dimensions. To match this requirement in our test, payloads smaller than the fixed size were padded with zero bits. As a result, the effective amount of embedded data remained constant regardless of actual payload size, producing consistent PSNR outcomes.

Table 4 supports the PSNR results by showing the SSIM value for the same experimental settings. The proposed method consistently achieves SSIM values that are either equal to or very close to 1.000, across all payload sizes and test images. This indicates that the visual distortion introduced by the watermark embedding remains minimal and largely imperceptible to the human visual system. At the largest payload size of 16 Kb, the proposed method still maintains SSIM values above 0.995 for most images, such as 0.996 for Airplane, 0.997 for Sailboat, and 0.996 for Right Eye. In contrast, the method by Zhou et al. [10] shows more noticeable degradation at higher payloads, with SSIM values declining to 0.983 for Airplane, 0.955 for Skin, and 0.951 for Polyp-2. Meanwhile, the method by Dong et al. [9] remains relatively stable but generally lower than the proposed method. These SSIM results reinforce the earlier PSNR findings, confirming that the proposed scheme is effective at preserving image quality even under increased payload.

4.2.3. Robustness evaluation

To assess robustness under extreme conditions, the proposed method was configured with a duplication factor of $R = 5$ and an embedding strength $\alpha = 25$ to improve resistance against both conventional distortions and adversarial attacks. Table 5 presents a comprehensive comparison of visual watermark detection results on the "Right Eye" image under various image processing and

Table 5. Visual watermark detection results on the Right Eye image under various attacks

Image	[9]	[10]	Prop.	Image	[9]	[10]	Prop.
							
No attack	**NC=1**	NC=0.980	**NC=1**	Sharpening	NC=0.979	NC=0.935	**NC=1**
							
JPEG2000	NC=0.962	NC=0.619	**NC=0.998**	Median Filter	NC=0.929	NC=0.612	**NC=0.946**
							
Salt & Pepper	NC=0.864	NC=0.668	**NC=0.971**	Contrast Adj.	NC=0.720	NC=0.350	**NC=1**
							
Gaussian Noise	NC=0.827	NC=0.642	**NC=0.984**	Resize	NC=0.885	NC=0.606	**NC=0.894**
							
Gaussian Blur	NC=0.758	NC=0.603	**NC=1**	Insertion	NC=0.742	NC=0.753	**NC=1**
Copy Attack	**NC=1**	NC=0.420	NC=0.961	BIM	NC=0.787	NC=0.559	**NC=0.902**
FGSM	NC=0.921	NC=0.676	**NC=0.999**	PGD	NC=0.834	NC=0.56	**NC=0.924**

adversarial attacks. Each row corresponds to a specific attack scenario, and for each method, the extracted watermark is shown alongside its NC value with respect to the original embedded watermark. The binary visualizations highlight incorrect bits in red, where red pixels denote bit errors in the extracted watermark. In contrast, both black and white pixels represent correctly extracted bits regardless of their binary value (0 or 1).

In the absence of any attack, all methods yield high NC values, with the proposed method and the Dong's method maintaining perfect recovery. Under mild distortions such as Sharpening, Contrast Adjustment, and Salt & Pepper noise, the proposed method still achieves perfect or near-perfect extraction ($NC \geq 0.971$), with minimal or no red pixels, indicating excellent robustness. Under stronger degradations such as Gaussian Noise, Resize, and Insertion, the method by Zhou et al. [10] shows significant degradation (e.g., $NC = 0.606$ for Resize and $NC = 0.753$ for Insertion), while the proposed method still maintains high accuracy, with substantially fewer bit errors compared to others. Importantly, the proposed method demonstrates superior resilience against adversarial attacks, such as FGSM ($NC = 0.999$), BIM ($NC = 0.902$), and PGD

Table 6. Extended robustness evaluation (1)

Attack Type	Airplane			House			Peppers			Sailboat			Skin		
	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.
PSNR (dB)	38.21	38.30	**42.1**	38.75	39.18	**42.1**	41.43	38.81	**42.19**	40.72	38.01	**42.1**	39.43	40.51	**42.09**
No Attack	**1.000**	0.998	**1.000**	**1.000**	0.997	**1.000**	**1.000**	0.987	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.981	**1.000**
JPEG2000	0.663	0.704	**0.888**	0.542	0.782	**0.832**	0.716	0.737	**0.921**	0.472	0.718	**0.819**	0.888	0.765	**0.999**
Salt & Pepper	0.706	0.794	**0.985**	0.774	0.82	**0.984**	0.79	0.756	**0.971**	0.792	0.801	**0.976**	0.949	0.793	**0.981**
Gaussian Noise	0.918	0.831	**0.995**	0.935	0.869	**0.992**	0.835	0.811	**0.993**	0.915	0.813	**0.991**	0.981	0.776	**0.993**
Gaussian Blur	0.020	0.676	**0.999**	0.028	0.687	**0.990**	0.124	0.755	**0.998**	0.036	0.721	**0.994**	0.692	0.657	**1.000**
Sharpen	0.993	0.888	**0.995**	0.993	0.889	**0.995**	0.930	0.764	**1.000**	0.964	0.821	**0.998**	0.993	0.930	**0.997**
Median Filter	0.479	0.746	**0.963**	0.284	0.729	**0.964**	0.606	0.799	**0.969**	0.281	0.782	**0.970**	0.880	0.722	**0.948**
Contrast Adj.	0.720	0.890	**0.835**	0.720	0.906	**0.985**	0.722	0.845	**1.000**	0.720	0.937	**0.998**	0.720	0.636	**0.998**
Resize	0.010	0.720	**0.892**	0.012	0.719	**0.879**	0.091	0.805	**0.885**	0.020	0.767	**0.886**	0.834	0.688	**0.895**
Insertion	0.898	0.989	**1.000**	0.815	0.997	**1.000**	0.810	0.977	**1.000**	0.878	0.993	**1.000**	0.767	0.981	**1.000**
Copy Attack	**1.000**	0.452	0.977	**1.000**	0.436	0.981	**1.000**	0.441	0.984	**1.000**	0.453	0.975	**1.000**	0.425	0.988
FGSM	0.993	0.862	**1.000**	**1.000**	0.851	**1.000**	0.967	0.858	**1.000**	0.998	0.843	**1.000**	0.993	0.842	**1.000**
BIM	0.904	0.634	**0.912**	**0.952**	0.661	0.924	0.885	0.611	**0.943**	**0.934**	0.641	0.932	**0.997**	0.606	0.932
PDG	**0.973**	0.667	0.941	**0.98**	0.657	0.955	0.934	0.656	**0.949**	**0.968**	0.627	0.956	**0.998**	0.638	0.955
Average	0.734	0.775	**0.956**	0.717	0.786	**0.963**	0.744	0.772	**0.972**	0.713	0.780	**0.964**	0.907	0.746	**0.978**

Table 7. Extended robustness evaluation (2)

Attack Type	Polyp-1			Polyp-2			Retina			Left Eye			Right Eye		
	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.
PSNR (dB)	38.77	41.24	**42.1**	37.98	39.77	**42.12**	41.91	41.27	**42.32**	38.81	42.27	**42.47**	37.77	42.22	**42.67**
No Attack	**1.000**	0.994	**1.000**	**1.000**	0.994	**1.000**	**1.000**	0.874	**1.000**	**1.000**	0.980	**1.000**	**1.000**	0.978	**1.000**
JPEG2000	0.397	0.67	**0.93**	0.859	0.773	**0.999**	0.773	0.655	**0.997**	0.962	0.619	**0.998**	0.959	0.681	**1.000**
Salt & Pepper	0.873	0.765	**0.968**	0.901	0.803	**0.975**	0.926	0.687	**0.953**	0.864	0.668	**0.971**	0.818	0.685	**0.942**
Gaussian Noise	0.947	0.816	**0.993**	0.953	0.839	**0.992**	0.961	0.690	**0.977**	0.827	0.642	**0.984**	0.680	0.667	**0.992**
Gaussian Blur	0.103	0.674	**0.995**	0.725	0.657	**1.000**	0.455	0.658	**1.000**	0.758	0.603	**1.000**	0.539	0.603	**0.997**
Sharpen	**0.991**	0.929	**0.991**	0.991	0.914	**0.993**	0.990	0.901	**1.000**	0.979	0.935	**0.999**	0.988	0.923	**0.997**
Median Filter	0.160	0.72	**0.967**	0.905	0.729	**0.954**	0.799	0.702	**0.965**	0.929	0.612	**0.946**	0.922	0.655	**0.952**
Contrast Adj.	0.720	0.643	**0.998**	0.720	0.836	**0.997**	0.720	0.524	**1.000**	0.720	0.350	**1.000**	0.720	0.398	**0.992**
Resize	0.102	0.713	**0.891**	0.871	0.738	**0.891**	0.555	0.691	**0.882**	0.885	0.606	**0.903**	0.637	0.635	**0.899**
Insertion	0.657	0.994	**1.000**	0.695	0.994	**1.000**	0.712	0.865	**1.000**	0.742	0.753	**1.000**	0.963	0.793	**1.000**
Copy Attack	**1.000**	0.429	0.971	**1.000**	0.434	0.979	**1.000**	0.423	0.987	**1.000**	0.420	0.961	**1.000**	0.435	0.979
FGSM	**1.000**	0.797	**1.000**	0.991	0.875	**1.000**	0.991	0.748	**1.000**	0.968	0.667	**1.000**	0.921	0.676	**0.999**
BIM	**0.989**	0.596	0.913	**0.967**	0.625	0.923	**0.997**	0.608	0.898	0.891	0.538	**0.91**	0.787	0.559	**0.902**
PGD	**0.997**	0.634	0.945	**0.99**	0.637	0.94	**0.996**	0.568	0.938	**0.934**	0.546	0.932	0.834	0.56	**0.924**
Average	0.710	0.741	**0.969**	0.898	0.775	**0.975**	0.848	0.685	**0.971**	0.890	0.639	**0.972**	0.841	0.661	**0.970**

(NC = 0.924), whereas the method in [9] and [10] experience considerable drops in NC (e.g., Zhou = 0.559 for BIM, 0.560 for PGD), and more pronounced bit errors. In the Copy Attack scenario, the proposed method still achieves reliable detection (NC = 0.961), whereas the method by Zhou et al. [10] fails to recover the watermark content (NC = 0.420).

More detailed results for the other images can be found in Table 6-7. Despite its overall effectiveness, the proposed method does not consistently outperform the baselines across all attack types. In copy attacks, the method by Dong et al. [9] frequently achieves perfect watermark recovery (e.g., NC = 1.000 on several images), while the proposed method records lower NC values. A similar pattern emerges under adversarial attacks like BIM and PGD, where the method by Dong et al. [9] occasionally surpasses the proposed method, for example, in Skin (PGD: 0.998 vs. 0.955) and Polyp-1 (PGD: 0.997 vs. 0.945). Nevertheless, the average NC scores for each image indicate that the proposed method consistently outperforms both the method in [9] and [10]. Across all ten images, the proposed method achieves higher average NC values, demonstrating greater overall robustness despite occasional weaknesses under specific attack scenarios.

The watermarking algorithm by Dong et al. [9] exhibits significant vulnerability to various types of attacks due to its strong dependence on the accuracy of singular values as the basis for watermark extraction. The method performs embedding in the low-frequency domain, specifically in the LL sub-band of the DWT transformed with DCT by modifying the singular values using a small

Table 8. Traceability detection metrics under various attack scenarios

Attack	Accuracy			Precision			Recall			F1		
	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.	[9]	[10]	Prop.
No Attack	0.000	**1.000**	**1.000**	0.000	**1.000**	**1.000**	0.000	**1.000**	**1.000**	0.000	**1.000**	**1.000**
JPEG2000	0.200	0.752	**1.000**	0.142	0.800	**1.000**	0.117	0.587	**1.000**	0.127	0.615	**1.000**
Salt & Pepper	0.108	0.862	**1.000**	0.204	**1.000**	**1.000**	0.180	0.770	**1.000**	0.191	0.797	**1.000**
Gaussian Noise	0.120	0.542	**1.000**	0.177	0.600	**1.000**	0.150	0.237	**1.000**	0.162	0.289	**1.000**
Gaussian Blur	0.350	0.528	**1.000**	0.133	0.400	**1.000**	0.610	0.213	**1.000**	0.077	0.236	**1.000**
Sharpen	0.840	**1.000**	**1.000**	0.169	**1.000**	**1.000**	0.136	**1.000**	**1.000**	0.150	**1.000**	**1.000**
Median Filter	0.256	0.832	**1.000**	0.129	0.800	**1.000**	0.106	0.720	**1.000**	0.116	0.746	**1.000**
Contrast	0.000	0.700	**1.000**	0.000	0.500	**1.000**	0.000	0.500	**1.000**	0.000	0.500	**1.000**
Resize	0.298	0.736	**1.000**	0.510	0.800	**1.000**	0.047	0.560	**1.000**	0.049	0.598	**1.000**
Insertion	0.378	**1.000**	**1.000**	0.026	**1.000**	**1.000**	0.020	**1.000**	**1.000**	0.023	**1.000**	**1.000**
Copy Attack	0.000	0.400	**1.000**	0.000	0.000	**1.000**	0.000	0.000	**1.000**	0.000	0.000	**1.000**
FGSM	0.090	0.906	**1.000**	0.190	**1.000**	**1.000**	0.147	0.843	**1.000**	0.159	0.866	**1.000**
BIM	0.100	0.400	**1.000**	0.228	0.000	**1.000**	0.159	0.000	**1.000**	0.180	0.000	**1.000**
PGD	0.124	0.400	**1.000**	0.249	0.000	**1.000**	0.200	0.000	**1.000**	0.218	0.000	**1.000**
Average	0.205	0.718	**1.000**	0.154	0.636	**1.000**	0.134	0.531	**1.000**	0.104	0.546	**1.000**

embedding factor α (~0.02). As a result, any modification to the image, such as contrast adjustment, gaussian noise, gaussian blur, or insertion attacks, directly affects the singular values of the image from which the watermark is to be extracted. This vulnerability arises because the extraction formula defined in Eq. (11) is highly sensitive to the accuracy of S_{WM}, which is derived from the DCT coefficients of the attacked image. Any distortion in these coefficients leads to deviations in the singular values from their expected values, thus impairing the watermark retrieval process.

The watermarking method proposed by Zhou et al. [10] exhibits significant vulnerabilities to various digital attacks due to several fundamental design weaknesses. In the case of copy attacks, the method is particularly susceptible as it adopts a deterministic approach that consistently selects image blocks with the highest texture values in a fixed order, without incorporating any randomization. This results in predictable embedding locations, making the watermark more susceptible to manipulation or removal through copy attack techniques. Additionally, the method is sensitive to embedding strength due to its reliance on the DC coefficient, where contrast adjustments or high-intensity disturbances can disproportionately affect the watermark embedded in the AC coefficients. Furthermore, the method lacks a watermark recovery mechanism, such as re-embedding the watermark across multiple locations, which could enhance robustness and increase the likelihood of successful extraction under adverse conditions.

Overall, the experimental results summarized in Table 6–7 demonstrates that the proposed method offers stronger robustness and reliability across a wide range of distortions. This resilience stems from three key design elements: (1) the use of logistic map-based randomization to determine embedding locations, which mitigates predictability and reduces susceptibility to spatial attacks such as copy attacks; (2) a repeated embedding strategy that increases redundancy and enhances the chances of successful watermark recovery under partial degradation; and (3) the application of a stronger embedding strength, which improves robustness against both conventional distortions and adversarial perturbations while preserving acceptable perceptual quality.

4.2.4. Traceability evaluation

Table 8 presents a detailed breakdown of accuracy, precision, recall, and F1-score, grouped by each type of attack scenario. Based on the experimental results shown in the table, the proposed watermarking method consistently outperforms the approaches by Dong et al. [9] and Zhou et al. [10] across all evaluation metrics accuracy, precision, recall, and F1-score under various attack scenarios. The proposed method achieves perfect scores (1.000) for all metrics under every type of attack. This indicates that the embedded identity fingerprints remain reliably detectable even after severe image distortions. In contrast, although the method by Zhou et al. [10] performs reasonably well under certain conditions (e.g., F1-score of 0.866 for FGSM), it suffers significant performance degradation under others, such as Gaussian Blur (F1 = 0.236) and BIM (F1 = 0.000). The method by Dong et al. [9], on the other hand, generally fails to identify the image source reliably, as reflected in its low average accuracy (0.205) and F1-score (0.104).

Despite its strengths in watermark extraction as seen in previous section, the Dong et al. [9] method suffers from a critical vulnerability: it cannot reliably

Table 9. Transaction performance evaluation

Metric	Measured Value
Total transactions	1000
Average time / transaction	8.866 seconds
Transaction throughput	43.91 transaction per second
Average gas / transaction	48380 gas units
Average gas price	26.69 Gwei
Total fee	1.291150 POL
Total fee (USD)	$0.2593

distinguish between different watermarks embedded in the same host image. This phenomenon, known as *cross-extraction*, undermines the security of the SVD-based embedding scheme. Cross-extraction occurs when a watermark can be extracted using an incorrect key particularly when different watermarks are embedded into the same host image. This issue arises from the method's reliance on the singular values of the LL component produced by DWT-DCT. The extraction formula in Eq. (11), is highly sensitive to variations in the host image parameters. When the same host image is used, the extracted S_{WM} values become highly similar across different watermarks, leaving S_H and the scaling factor α both stored in the key as the main distinguishing elements, which remain nearly identical. Moreover, watermark reconstruction via inverse SVD is dominated by the spatial structures encoded in matrices U and V, rather than the embedded singular values. Consequently, even an incorrect key may produce a visually plausible watermark pattern. This weakness violates two fundamental principles of secure watermarking: *uniqueness* and *non-repudiation* resulting in the method in [9] is unreliable for applications requiring ownership verification or digital content authentication.

Overall, the average metric scores for the proposed method reach 1.000, significantly surpassing those of the method by Zhou et al. [10] (e.g., average F1-score = 0.546) and the method by Dong et al. [9] (average F1-score = 0.104). This demonstrates the high reliability of the proposed system in tracing image distribution sources with precision, even under extreme distortion conditions.

4.2.5. Evaluation of smart contract performance

A total of 1,000 transactions were submitted, simulating the distribution of 10 host images to 100 different recipients each. Each transaction recorded metadata including the identities of the data owner and requester, perceptual hash, fingerprint, and IPFS references. The benchmark results are summarized as follows.

Table 10. Performance evaluation of IPFS

Parameter	Upload	Retrieve
Success	1000	10000
Failed	0	0
Avg. Latency	1.0087 s	0.8155 s
Availability	100%	100%

The average processing time per transaction was recorded at 8.866 seconds, with a throughput of 43.91 transactions per second in the Polygon Mainnet environment. In terms of gas consumption, each transaction required an average of 48,380 gas units, with an average gas price of 26.69 Gwei. When executed on the Polygon network, the total gas fee for all transactions amounted to approximately 1.291150 POL (Polygon). Given the price of POL on July 9, 2025, which was $0.2008, the estimated total monetary cost for these 1,000 transactions was only $0.2593. This demonstrates the scalability and cost-efficiency of deploying the proposed traceability mechanism on Polygon Mainnet, making it highly suitable for real-world high-frequency image sharing applications.

4.2.6. IPFS performance evaluation

The performance evaluation of IPFS was conducted by measuring both upload and retrieval processes on the configured server described in Section 5.1.6. As summarized in Table 10, a total of 1,000 image delta files were uploaded to the IPFS node, each with a success rate of 100%. The average latency for uploading was measured at 1.0087 seconds, and no upload failures occurred. The average file size of the image delta files was 106.30 KB.

For the retrieval phase, each uploaded file was accessed 10 times, resulting in 10,000 retrieval operations. Similar to the upload phase, all retrievals were completed successfully with zero failure and 100% availability. The average retrieval latency was slightly lower at 0.8155 seconds, indicating efficient content addressing and data propagation within the IPFS network under repeated access conditions.

These results demonstrate that even in a resource-constrained VPS environment, IPFS provides high availability, stable response times, and consistent performance when used as a decentralized storage layer for traceable image watermarking systems.

4.2.7. Threat model and security analysis

4.2.7.1. Threat model

In the proposed system, we assume a threat model

Table 11. Attack scenarios and mitigations

Attack Type	Description	Mitigation Strategy
Key Compromise	Attacker obtains a requester's private key and accesses encrypted IPFS data (e.g., extraction key).	Even with access to the reference DCT coefficients (stored via IPFS), extraction remains infeasible without knowing the logistic map key that determines the block embedding order.
Unauthorized Watermark Extraction	Extraction attempt without knowledge of logistic map parameters.	The block order is randomized using a logistic map with secret parameters, which are not stored on-chain. Extraction fails without these keys.
Metadata Poisoning	Attacker attempts to inject false ownership claims on the blockchain.	The watermark embedded in each image is a verifiable ECDSA signature to the data owner, and the ECDSA is tied to specific content and requester identity. Verification ensures authenticity and binds the fingerprint to its rightful owner.
Smart Contract Exploitation	Malicious users submit arbitrary or fake records to the contract.	The smart contract accepts open inserts, but fake records are ineffective: traceability is based on image-embedded ECDSA signatures, which can be cryptographically verified against the claimed owner.
Watermark Copy Attack	Attacker averages multiple watermarked images to suppress the embedded watermark signal.	Watermark bits are redundantly embedded across multiple locations using logistic-map-based block selection. This redundancy and randomness reduce the effectiveness of averaging-based removal.

involving three types of entities: *(a) Honest Participants*: legitimate data owners and requesters who comply with the protocol, *(b) Malicious Requesters*: legitimate recipients who attempt to redistribute or reverse-engineer the watermark., and *(c) External Adversaries*: attackers who intercept images, probe smart contracts, or attempt to manipulate blockchain records.

The system assumes secure key storage by legitimate users and relies on cryptographic mechanisms, particularly ECDSA and logistic-based randomization to protect against forgery and unauthorized trace reconstruction.

4.2.7.2. Attack scenarios and mitigation strategies

Table 11 presents a summary of the key attack scenarios that may compromise the integrity, authenticity, or traceability of the proposed system. Each row outlines a distinct threat, its corresponding adversarial objective, and the defense mechanism employed within the system design. As shown in the table, the use of randomized logistic map parameters ensures that unauthorized watermark extraction is computationally infeasible, even if IPFS-stored references are accessible. Furthermore, embedded ECDSA signatures act as tamper-proof verifiable claims, which effectively defend against metadata poisoning and ownership forgery.

4.2.7.3. Security properties

The proposed system provides several security guarantees essential for secure and traceable image sharing. *Confidentiality* is maintained by encrypting both the image delta and the watermark extraction key using the requester's public key before storing them on IPFS. Without the corresponding private key, unauthorized parties cannot recover the original image or the embedded fingerprint. *Integrity and authenticity* are enforced through ECDSA signatures embedded in the image, which are generated using the data owner's private key. These signatures are verifiable against the public key stored on the blockchain, ensuring that the fingerprint cannot be forged or tampered with. *Availability* is ensured by utilizing decentralized infrastructures: IPFS for storing watermark-related files and the Polygon mainnet for recording transaction metadata. Our deployment and testing show 100% availability in both upload and retrieval processes on IPFS, and Polygon provides high transaction uptime and accessibility, demonstrating strong operational reliability. Lastly, *transparency* is ensured through the use of a public blockchain and IPFS, allowing all metadata and distribution records to be independently audited without relying on a centralized authority.

5. Conclusions

This paper presented a novel image data sharing framework that integrates reversible robust watermarking with blockchain technology to enable traceable and secure image distribution. By embedding a unique 1024-bit ECDSA-based digital

fingerprint that identifies both the image and the requester, the proposed method ensures accountability in every distribution event. The use of logistic map-based randomized embedding, perceptual weighting through CSF, and redundant bit insertion significantly enhances both robustness and imperceptibility, achieving average PSNR above 50.29 dB for 16 Kb payload and SSIM values consistently close to 1.000.

Extensive experiments under various distortion and adversarial scenarios confirmed that the proposed scheme outperforms the method by Dong et al. [9] and the method by Zhou et al. [10] in terms of watermark recovery accuracy (average NC = 0.97) and traceability (F1-score = 1.000).

Beyond empirical performance, the method is supported by several theoretical design elements that differentiate it from existing schemes. The use of logistic map-based block randomization secures the watermark location, making the scheme resistant to spatial and averaging attacks. The redundant embedding of watermark bits increases resilience against both noise and adversarial perturbations. In addition, the contrast sensitivity-based perceptual weighting balances robustness and imperceptibility by adapting the embedding strength to human visual tolerance.

Compared to existing methods by Dong et al. [9] and Zhou et al. [10], the proposed method offers clearer advantages. The method by Dong et al. [9] lacks traceability granularity and is vulnerable to cross-extraction due to its dependence on singular values. The method by Zhou et al. [10], although robust to some attacks, uses deterministic block selection and does not support reversibility, limiting its applicability in sensitive domains. In contrast, our approach combines traceability, reversibility, and verifiability, a combination not simultaneously addressed in prior work.

Furthermore, deployment on the Polygon Mainnet and IPFS demonstrated the practicality of the proposed system in real-world environments, with low-cost, high-throughput smart contract interactions and reliable decentralized storage performance. The system demonstrates effective mitigation strategies against key compromise, unauthorized watermark extraction, metadata poisoning, and watermark copy attacks, as outlined in Table 11. Overall, the proposed approach offers a practical and verifiable mechanism for image provenance tracking and copyright management, especially in use cases that require both traceability and reversibility, such as medical image distribution and digital forensics.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, IKAAP; methodology, IKAAP; software, IKAAP; validation, IKAAP, BAP, and HS; resources, BAP and HS; writing—original draft preparation, IKAAP; writing—review and editing, IKAAP, BAP and HS; visualization, IKAAP; supervision, BAP and HS; project administration, BAP and HS; funding acquisition, BAP and HS.

Acknowledgments

This research is funded by the Indonesian Endowment Fund for Education (LPDP) on behalf of the Indonesian Ministry of Education, Culture, Research, and Technology and managed under the INSPIRASI Program Grant No. 2965/E3/AL.04/2024.

References

[1] I. Gupta, A. K. Singh, C.-N. Lee, and R. Buyya, "Secure Data Storage and Sharing Techniques for Data Protection in Cloud Environments: A Systematic Review, Analysis, and Future Directions", *IEEE Access*, Vol. 10, pp. 71247–71277, 2022, doi: 10.1109/access.2022.3188110.

[2] Y.-H. Chen and M.-C. Huang, "Fine-Grained Encrypted Image Retrieval in Cloud Environment", *Mathematics*, Vol. 12, No. 1, p. 114, Dec. 2023, doi: 10.3390/math12010114.

[3] W. Wang, "Secure Image Retrieval and Sharing Technologies for Digital Inclusive Finance: Methods and Applications", *Trait. Signal*, Vol. 40, No. 5, pp. 2079–2086, Oct. 2023, doi: 10.18280/ts.400525.

[4] Z. Lu, J. Wei, C. Li, J. Zhai, and D. Tong, "Robust copyright tracing and trusted transactions using zero-watermarking and blockchain", *Multimed. Tools Appl.*, Vol. 84, No. 12, pp. 10687–10724, 2024, doi: 10.1007/s11042-024-19325-2.

[5] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", *Scientific Research*, 2008.

[6] Z. B. Faheem *et al.*, "Image Watermarking Scheme Using LSB and Image Gradient", *Appl. Sci.*, Vol. 12, No. 9, p. 4202, Apr. 2022, doi: 10.3390/app12094202.

[7] A. O. Mohammed, H. I. Hussein, R. J. Mstafa, and A. M. Abdulazeez, "A blind and robust color image watermarking scheme based on

DCT and DWT domains", *Multimed. Tools Appl.*, Vol. 82, No. 21, pp. 32855–32881, Sep. 2023, doi: 10.1007/s11042-023-14797-0.

[8] R. Y. Abadi and P. Moallem, "Robust and optimum color image watermarking method based on a combination of DWT and DCT", *Optik*, Vol. 261, p. 169146, 2022, doi: 10.1016/j.ijleo.2022.169146.

[9] Y. Dong, R. Yan, Q. Zhang, and X. Wu, "A Hybrid Domain Color Image Watermarking Scheme Based on Hyperchaotic Mapping", *Mathematics*, Vol. 12, No. 12, p. 1859, 2024, doi: 10.3390/math12121859.

[10] Q. Zhou, S. Shen, S. Yu, D. Duan, and Y. Yuan, "An Image Watermarking Algorithm Based on Embedding Pixel Values Directly with DCT and Block Selection", *Circuits Syst. Signal Process.*, 2025, doi: 10.1007/s00034-025-03149-y.

[11] Q. Li, X. Wang, and Q. Pei, "A robust reversible watermarking scheme overcomes the misalignment problem of generalized histogram shifting", *Multimed. Tools Appl.*, Vol. 82, No. 5, pp. 7207–7227, 2023, doi: 10.1007/s11042-022-13611-7.

[12] M. Fan, S. Zhong, and X. Xiong, "Reversible Data Hiding Method for Interpolated Images Based on Modulo Operation and Prediction-Error Expansion", *IEEE Access*, Vol. 11, pp. 27290–27302, 2023, doi: 10.1109/ACCESS.2023.3258461.

[13] L. Tanwar and J. Panda, "Hybrid reversible watermarking algorithm using histogram shifting and pairwise prediction error expansion", *Multimed. Tools Appl.*, Vol. 83, No. 8, pp. 22075–22097, 2023, doi: 10.1007/s11042-023-15508-5.

[14] A. Garba *et al.*, "A digital rights management system based on a scalable blockchain", *Peer--Peer Netw. Appl.*, Vol. 14, No. 5, pp. 2665–2680, 2021, doi: 10.1007/s12083-020-01023-z.

[15] J. Yun, X. Liu, Y. Lu, J. Guan, and X. Liu, "DRPChain: A new blockchain-based trusted DRM scheme for image content protection", *PLOS ONE*, Vol. 19, No. 9, p. e0309743, 2024, doi: 10.1371/journal.pone.0309743.

[16] X. Yi, Y. Zhou, Y. Lin, B. Xie, J. Chen, and C. Wang, "Digital rights management scheme based on redactable blockchain and perceptual hash", *Peer--Peer Netw. Appl.*, Vol. 16, No. 5, pp. 2630–2648, 2023, doi: 10.1007/s12083-023-01552-3.

[17] R. Ma, L. Zhang, Q. Wu, Y. Mu, and F. Rezaeibagha, "BE-TRDSS: Blockchain-Enabled Secure and Efficient Traceable-Revocable Data-Sharing Scheme in Industrial Internet of Things", *IEEE Trans. Ind. Inform.*, Vol. 19, No. 11, pp. 10821–10830, 2023, doi: 10.1109/TII.2023.3241618.

[18] Z. Wang and S. Guan, "A blockchain-based traceable and secure data-sharing scheme", *PeerJ Comput. Sci.*, Vol. 9, p. e1337, 2023, doi: 10.7717/peerj-cs.1337.

[19] P. Verma, V. Tripathi, and B. Pant, "MRDACE: An Intelligent Architecture for Secure Sharing and Traceability of the Medical Images and Patients' Records", *ACM Trans. Comput. Healthc.*, Vol. 6, No. 3, pp. 1–21, 2025, doi: 10.1145/3712708.

[20] C. Lai, Z. Ma, R. Guo, and D. Zheng, "Secure medical data sharing scheme based on traceable ring signature and blockchain", *Peer--Peer Netw. Appl.*, Vol. 15, No. 3, pp. 1562–1576, 2022, doi: 10.1007/s12083-022-01303-w.

[21] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete Cosine Transform", *IEEE Trans. Comput.*, Vol. C–23, No. 1, pp. 90–93, 1974, doi: 10.1109/T-C.1974.223784.

[22] J. Mannos and D. Sakrison, "The effects of a visual fidelity criterion of the encoding of images", *IEEE Trans. Inf. Theory*, Vol. 20, No. 4, pp. 525–536, 1974, doi: 10.1109/TIT.1974.1055250.

[23] G. Sharma, *Digital Color Imaging Handbook*, CRC Press, 2002.

[24] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System", *arXiv Preprint, arXiv:1407.3561*, 2014, doi: 10.48550/arXiv.1407.3561.

[25] D. Johnson, A. Menezes, and S. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)", *Int. J. Inf. Secur.*, Vol. 1, No. 1, pp. 36–63, 2001, doi: 10.1007/s102070100002.

[26] M. Lochter and B. Merkle, "Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation", *Internet Engineering Task Force (IETF)*, 2010.

[27] USC Signal and Image Processing Institute, "USC-SIPI Image Database", *University of Southern California*, 1977.

[28] M. M. Fraz *et al.*, "An Ensemble Classification-Based Approach Applied to Retinal Blood Vessel Segmentation", *IEEE Trans. Biomed. Eng.*, Vol. 59, No. 9, pp. 2538–2548, 2012, doi: 10.1109/tbme.2012.2205687.

[29] K. Pogorelov *et al.*, "KVASIR: A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection", In: *Proc. of the 8th ACM on Multimedia Systems*

Conference, Taipei Taiwan, pp. 164–169, 2017, doi: 10.1145/3083187.3083212.

[30] International Skin Imaging Collaboration (ISIC), "ISIC Skin Image Dataset", 2020. [Online]. Available: https://www.isic-archive.com

[31] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss, "The FERET database and evaluation procedure for face-recognition algorithms", *Image Vis. Comput.*, Vol. 16, No. 5, pp. 295–306, 1998, doi: 10.1016/S0262-8856(97)00070-X.