electronics

Article

Enhancing Steganography through Optimized Quantization Tables

Rasa Brūzgienė *, Algimantas Venčkauskas †, Šarūnas Grigaliūnas † and Jonas Petraška

Department of Computer Sciences, Kaunas University of Technology, Studentu Str. 50, 51368 Kaunas, Lithuania; algimantas.venckauskas@ktu.lt (A.V.); sarunas.grigaliunas@ktu.lt (Š.G.); xaresas@gmail.com (J.P.)

* Correspondence: rasa.bruzgiene@ktu.lt

† These authors contributed equally to this work.

Abstract: This paper addresses the scientific problem of enhancing the security and capacity of steganographic methods for protecting digital media. The primary aim is to develop an advanced steganographic technique that optimizes quantization tables to surpass the traditional F5 algorithm in terms of security, capacity, and robustness. The novelty of this research lies in the introduction of the F5A method, which utilizes optimized quantization tables to significantly increase the capacity for concealed information while ensuring high-quality image retention and resistance to unauthorized content recovery. The F5A method integrates cryptographic keys and features to detect and prevent copyright infringement in real time. Experimental evaluations demonstrate that the F5A method improves the mean square error and peak signal-to-noise ratio indices by 1.716 and 1.121 times, respectively, compared to the traditional F5 algorithm. Additionally, it increases the steganographic capacity by up to 1.693 times for smaller images and 1.539 times for larger images. These results underscore the effectiveness of the F5A method in enhancing digital media security and copyright protection.

Keywords: data security; privacy; digital watermarking; steganography; forensics and detection

check for updates

Citation: Brūzgienė, R.; Venčkauskas, A.; Grigaliūnas, Š.; Petraška, J. Enhancing Steganography through Optimized Quantization Tables. *Electronics* **2024**, *13*, 2415. https://doi.org/10.3390/electronics13122415

Academic Editor: Simeone Marino

Received: 13 May 2024
Revised: 6 June 2024
Accepted: 18 June 2024
Published: 20 June 2024

1. Introduction

The advent of the digital revolution has facilitated substantial economic expansion and democratized media production, allowing individuals and businesses to utilize internet platforms as significant sources of income. The internet has enabled unprecedented potential for digital media creation, distribution, and consumption. However, the convenience and widespread availability of digital media have also increased the risks associated with protecting them from copyright infringement [1]. This issue not only has legal implications but also significant economic consequences. The core of digital media security issues lies in the unauthorized reproduction and alteration of content, commonly referred to as "piracy". This illicit activity persists as a significant challenge, affecting creators and copyright holders across various media, including video, audio, text, and more. Recent studies, such as those referenced in [2], estimate that over three billion individuals engage in the unauthorized downloading of recorded music monthly, underscoring the vast scale of this problem.

Video and images disseminated on social media platforms are particularly popular with younger demographics, as well as older individuals. Advertising companies are increasingly recognizing the growing popularity of social networks as a means to contact their target demographic and promote their products. They offer a generous compensation package based on viewership and other factors, which motivates content creators to produce more captivating and unique material. Nevertheless, not all content creators exhibit equitable behavior. Frequently, individuals create profiles on social networks with the intention of generating income by uploading pilfered content. Copyright laws are violated when you plagiarize someone else's content and claim it as your own [3]. Although digital media security solutions are present on social networking platforms to detect and alert content authors about duplicate content, they generally do not prevent content misappropriation, such as copying a post from one platform and uploading it on another.

As long as uploading and sharing data does not violate copyright laws, it is legally permissible. However, instances of unauthorized copying and distribution of files are frequently observed [4]. In this case, the copyright violator uploads the file to a file upload platform, then shares the download URL on the internet for others to access. Individuals who come across this hyperlink have the ability to acquire the unlawfully obtained information without being aware of its illicit origin. This distribution of files constitutes a breach of digital file security as it infringes upon the copyright of the rightful owners. It may potentially be considered a copyright violation to distribute a hyperlink to anything. This pertains to instances where you incorporate digital media content, like a photograph or video, into your personal blog or social media account. While the provided link directs you to the genuine author's page, sharing another author's photographs or movies with other parties without their permission is also a violation of copyright laws. While not directly related to digital media safety, this issue is primarily about piracy. The harm occurs when audiences outside the intended region access the content, resulting in a loss of income for the author. Obtaining the author's permission is necessary in order to mention their name and include complete links to their social media profiles when sharing the content they have produced.

Given these challenges, enhancing digital media protection has become imperative. It is almost impossible to trace every illegal use on the internet, so individuals accused of piracy are often invisible in court due to the unreasonable time and financial costs involved. Encrypting sensitive data can help prevent unauthorized use. Cryptography enables the encryption of a wide range of files. Cryptographic encryption restricts unauthorized downloads of copyrighted content to the author's defined environment, which can decode the file upon authorization. Cryptographic signatures can also be used to verify copyright [5]. To protect digital content from unauthorized use, cryptographic signatures establish ownership.

Tags are a simple yet effective approach to securing content. Tags are widespread on online videos and images. The organization's emblem often appears as a hard-to-translucent layer over photographs and videos. Viewers avoid duplicating and unlawfully using tagged content because it makes them uncomfortable. This method protects digital media files from small-scale piracy, but more advanced users can get around it by using graphic design tools to remove the markings. On the other hand, Blockchain technology tags digital media files and preserves ownership rights via customized contracts [6]. Blockchain contracts are publicly available but unchangeable. However, if the content's copyright changes or infringes on another work's copyright, the Blockchain entry will not be deleted, and the content's appropriation information will be kept.

The current methods, including traditional digital rights management systems [7], are often circumvented, leaving content vulnerable. The advancement of steganography—the art of hiding information within other information—is one promising avenue. Digital watermarks track copyrighted content by employing steganographic methods [8]. Digital watermarks can prove copyright and track unlawful distributors, but they do not immediately prevent content copying. Copyists may not see digitally watermarked work, making it easier to track and take legal action to preserve copyright [9]. Conversely, steganography can serve as a covert mechanism to embed copyright data within the media itself, making unauthorized use more detectable and traceable without affecting the quality or integrity of the content [10]. Steganography is focused on the concealment of a message within another medium, ensuring that the existence of the embedded information remains undetected except for the intended parties [11]. However, the application of steganography brings its own set of challenges. While there are many steganography-driven solutions, most of them only provide a legal basis for compensation. The protection of digital content against unauthorized use is still an open area for research. In relation to this, the main aim of this paper is to propose a cutting-edge steganographic method called F5A for enhanced security of digital media, particularly focusing on digital images. The authors' main contributions are as follows:

- A new steganographic method called F5A has been proposed, which is based on the F5 algorithm and extends its capacity by using generated quantization tables, as well as adding new features such as protection against reconstruction of the carrier's content and notification of copyright infringement during decryption.
- A copyright protection solution capable of identifying infringing content and notifying the copyright owner.
- Implemented cryptography-based protection of identified content against unauthorized use.

The proposed method allows for hiding of as much information as possible, as the original image hides the entire image copy, which must remain of the best possible quality. It can also be easily combined with an embedded cryptographic key to encrypt the information in order to reliably protect the hidden content from disclosure. The F5A method is resistant to file modifications such as stretching, cutting, compression, or other processing. This approach not only facilitates the tracing and identification of unauthorized content distribution but also guarantees the preservation of the original media's quality.

This paper is organized in the following structured format. Section 2 provides a comprehensive review of existing steganographic methods and algorithms utilized in copyright protection. The proposed F5A method is detailed in Section 3, along with the corresponding experimental methodology and analysis of the results presented in Section 4. Section 5 elucidates the merits and limitations of the proposed approach. Finally, Section 6 summarizes the main findings and delineates directions for future research.

2. Related Works on Image Steganography

Steganography is a way of hiding sensitive information in plain sight by deceiving the observer of an image or other file [12]. This method of information concealment enables private communication between individuals who understand the hidden messages and their interpretation. Steganography is not a form of cryptography; instead, cryptography and other techniques are used to hide information, depending on the size and type of data file chosen. The chosen format will determine the maximum amount of encrypted information and the methods used to conceal it.

This section reviews various traditional and contemporary steganographic methods, emphasizing their applicability and limitations in digital copyright protection. It compares different techniques' effectiveness and introduces the concept of embedding hidden information in digital media, which serves as a prelude to the innovative F5A method detailed later. The section aims to contextualize the F5A method within the broader landscape of steganography and digital rights management, highlighting the ongoing challenges and developments in the field .

Image steganography is one of the most popular types of steganography [13]. Sensitive information posted in images can easily circulate on social networks and forums without arousing much suspicion. The vast amount of visual information on the internet allows secret messages to go unnoticed, and by blending in with regular content, they can remain hidden for a very long time. To hide a message in an image without causing a noticeable difference to the original version of the image, it is possible to cover the image with noise, use color variations, or use other techniques to hide the message. The authors in [14] explore multiple deep learning techniques applied to image steganography, encompassing conventional approaches, CNN-based strategies, and GAN-based techniques. Additionally, they detail the data sets, experimental configurations, and evaluation criteria frequently utilized in this domain.

Data security is ensured in [15] through an innovative steganography technique that incorporates generative adversarial networks. This method primarily focuses on embedding concealed information during the creation of art-style images, thereby making it challenging for adversaries to detect steganographic content. It is capable of embedding up to three bits per pixel in a color image, significantly increasing the amount of secret data that can be hidden within a single image. However, the GAN-based approach entails intricate neural network architectures and training processes, necessitating considerable computational resources and time.

The computational resource-constrained hybrid security system proposed in [16] for encrypting color images employs a combination of SHA256-MD5 hash functions, DNA permutation, and diffusion with a 5D hyper-chaotic system to enhance security. This approach specifically shows high resistance to noise and differential attacks while maintaining speed due to its lower computational complexity.

The proposed solutions in [17,18] enhance RGB color image encryption by combining the spatial and transformation approaches using a hybrid optimization algorithm that integrates the Salp Swarm algorithm and the arithmetic optimization algorithm. This gives higher security through complex chaotic systems, improved robustness against various attacks, optimized encryption parameters, and the integration of dual encryption approaches. However, it is more focused on encryption rather than embedding capacity.

The least significant bit (LSB) is the most popular method for steganography of visual material. This method embeds information into an image using the least significant bits of the image, and the least significant bits of an image are those whose value, when changed by the image observer, makes the processed image least different from the original. Pixel Value Differencing (PVD) with a quantized range table approach is one of the examples of this type of image steganography. The authors in [19] proposed a technique that takes advantage of humans' limited ability to discern minor changes in pixel values, particularly in areas of high visual complexity. This methodology uses the quantized range table method to improve resilience against unauthorized detection and decoding while maintaining the visual integrity of the stego-image. However, the approach has drawbacks, including the possibility of being vulnerable to advanced steganalysis tools that rely on knowledge of PVD-based steganography properties. The work emphasizes the need for additional research to fine-tune the technique, particularly to expand its use to 3D images and improve its security features to prevent advanced analytical attacks. Other researchers employ a pixel locator sequence (PLS) combined with AES encryption [20]. The PLS distributes data randomly across image pixels, enhancing security. However, the space-efficient nature of conveying meta-data (PLS) and the need to balance security and space efficiency pose limitations in this context.

The authors in [21] improve the LSB with matching revisited technique by efficiently hiding messages with more than two bits while minimizing alterations to the image. However, this method could lead to increased computational demands due to the complexity of the operations required for embedding and retrieving messages. Additionally, despite its improved capacity, the technique could remain vulnerable to sophisticated steganalysis methods if not executed with precision, particularly as detection technologies for steganography continue to advance. In [22], a changed multiway pixel-value differencing method using universal quantization ranges to make image steganography better was explained. This approach boosts embedding rates while maintaining high image quality. However, its drawbacks include additional complexity and potential security trade-offs under certain structural analysis scenarios.

Only the steganography of 24-bit or black-and-white images typically uses the masking and filtering method. Masking and filtering techniques are similar in principle to watermarks or a specific scribble on an image. This method, unlike LSB, visually represents the changes, making it possible for the image's observer to notice them, but proper use would make it difficult to discern the differences. However, this method has advantages over LSB in that the masking and filtering method makes the image resistant to compression, cropping, or other modifications to the image size. It is more suitable for the steganography of JPEG files than the LSB method because it visually modifies the image and makes it resistant to compression. The study in [23] introduces an automated method for categorizing digital images by noise levels using the Canny filter, aimed at optimizing the selection of images for steganographic data embedding. However, the approach may face challenges in accurately assessing noise levels, potentially impacting the efficacy of steganographic techniques.

The transformation method is more complicated than the previous methods because it uses complex mathematical functions, such as the discrete cosine transform. The JPEG compression algorithm employs the discrete cosine transform (DCT). This algorithm can compress not only the image but also the generated secret message together, thus hiding the message information in the already compressed file. The authors in [24] scattered the cover image to broadly disperse energy, increasing the number of DCT coefficients available for embedding data. The quantization table values then modify the underlying capacity of each DCT block, yielding varying results across different tables.

The proposed approach in [25] improves JPEG steganography by using a quantized Gaussian model that takes into account embedding costs or residual variances measured in both the spatial and DCT domains. It greatly enhances security for various payloads. However, because of its advanced statistical computations, the method may add increased complexity and, therefore, reduced practicality for real-time applications.

The study in [26] presents the robust adaptive steganography method based on post-processing and accurate dither modulation. The method specifically caters to images of exceptional quality. This technique leverages accurate dither modulation to minimize the scope of alterations and utilizes postprocessing to improve robustness against JPEG recompression. However, these essential alterations have the potential to increase security risks.

Digital file formats, including text file formats, use steganographic coding algorithms, also known as stegosystems, to embed various types of data, such as code, images, or audio. Steganographic encoding can be applied to almost all file formats, but it is more practical to apply steganographic encoding to some formats than others. The paper in [27] presents a method to improve the JPHide steganography system by advancing the techniques for retrieving the check matrix and shuffling key. It achieves this by analyzing embedding ratios and distinguishing genuine from counterfeit keys using the distribution of bit sequences. Nonetheless, the approach may face difficulties in effectively retrieving keys when embedding ratios are at their extremes, potentially reducing the robustness of the stegosystem under certain conditions.

Jsteg replaces the least significant bits of the DCT coefficients with the secret message bits, omitting those coefficients with values of 0 or 1. The algorithm reads the file sequentially and does not allow for random bit selection. Before embedding, a steganographic key encrypts the message. The research in [28] adds a robustness cost function to improve the security of JPEG steganography. This makes the message more reliable after it has been compressed. However, this method slightly sacrifices security effectiveness. Another study in [29] focuses on employing co-frequency sub-image filtering to predict cover images for payload location. This method improves accuracy in determining stego positions in JPEG images. Using multiple JPEG compression factors can lead to inaccuracies in predicting cover pictures, which is a major issue.

The JPHide algorithm also relies on replacing the least significant bits of the DCT coefficients, but it does not do so consistently. Instead, it employs a fixed table to determine the next coefficient to change. The tool also employs a semi-random number generator to omit some of the coefficients, with the probability of omission varying based on the number of inserted bits and the remaining bits. The study in [30] looks at how resistant JPHide steganography is to different image processing changes. It shows how hard it is to find JPHide in all of these different processing situations. Limitations include the method's decreased effectiveness due to variations in JPEG compression and image enhancement, which complicates the consistent detection of hidden data.

The OutGuess steganographic encoding algorithm is another JPEG stegosystem that uses least significant bit encoding for DCT coefficients. In contrast to the two previous algorithms, OutGuess chooses the coefficients randomly using semi-random number generation with a user-supplied password. In [31], the authors focus on utilizing the stability characteristics of JPEG-quantified DCT parameters. Their research provides a new embedding solution based on the position consistency of the last nonzero DCT coefficients, which is particularly susceptible to JPEG compression. Limitations include potential security trade-offs, as the process may be more detectable using specific steganalysis tools.

The F5 algorithm can be seen as an improved version of the stegosystems described above. It introduces a number of new ideas, including the use of matrix coding to reduce the number of changes required. Additionally, it incorporates permutation switching to guarantee an even distribution of modifications throughout the data. F5 reduces the propagation of steganographic information through the carrier. The method described in [32] aims to get the check matrix and shuffling key of F5 steganography by guessing the embedding ratios and telling the difference between real and fake keys by looking at the bit sequence distributions. Notably, it may encounter challenges in accurately recovering keys at particularly high or low embedding ratios.

The study in [33] improves the F5 steganography method by making it easier to obtain the check matrix and shuffle the key. It does this by using estimates of embedding ratios and bit sequence distributions to tell the difference between real and fake keys. The method's reduced efficacy at very high or low embedding ratios is the primary limitation noted, potentially compromising key recovery accuracy.

The research in [34] examines four distinct image steganography techniques—LSB embedding, pseudo-random LSB (PRLSB) embedding, EzStego, and F5—applied to five images and evaluated using various metrics, including the Chi-Square Test, peak signal-to-noise ratio (PSNR), and computational complexity. The findings indicate that the F5 algorithm outperforms the others in most assessments, except in the structural similarity index test, where LSB embedding shows the best results. The study highlights the variable efficacy of each method depending on the context, underscoring the absence of a universally optimal approach in image steganography.

3. F5A Steganographic Method

This section details the newly developed F5A steganographic method, which improves upon the traditional F5 algorithm by utilizing optimized quantization tables. It describes the methodological enhancements, including increased steganographic capacity and resistance to unauthorized decryption, which are critical for effective digital copyright protection. This segment serves as a technical core of the document, explaining the innovations and cryptographic integrations of the F5A method .

The F5 method is characterized by a reduction in steganographic information propagation and increased resilience to file modifications such as stretching, cutting, compression, or other processing. However, it has a low capacity for embedded data. The F5A, also known as the F5 Advanced, steganographic method aims to ensure a large amount of embeddable secret information, not to maintain image quality using standard JPEG quantization tables. The proposed F5A steganography method provides copyright protection and can identify copyright-infringing content (stolen image). The F5A method, once implemented, will function as a search engine for copyrighted content, scrutinizing the user's opened web page for potential infringements and promptly notifying the user. The list of copyright documents is stored in the API, so when images are found, the data of the found images is sent to the API, and a response is received about the infringement status of the image. When the search engine detects a copyright infringement, it notifies the author via email or message, ensuring the infringement data remain current.

Optimized quantization tables can be employed to enhance the capacity of DCT-based steganographic methods, thereby creating more space for the insertion of secret information following the discrete cosine transform. This is possible because a significant proportion of the coefficients remain at 0 values after the DCT. The original F5 method does not utilize these 0 values to encode secret information. However, by increasing the number of non-zero values through the use of a modified quantization table, the capacity of the steganographic algorithm can be significantly enhanced. This principle forms the foundation of the F5A method, which leverages this increased capacity to improve steganographic performance. Given the F5 algorithm's capacity constraints, the proposed F5A solution generates optimal quantization tables using a selectable compression ratio to increase the hidden information's capacity.

Figure 1 shows the encoding steps of the F5A method:

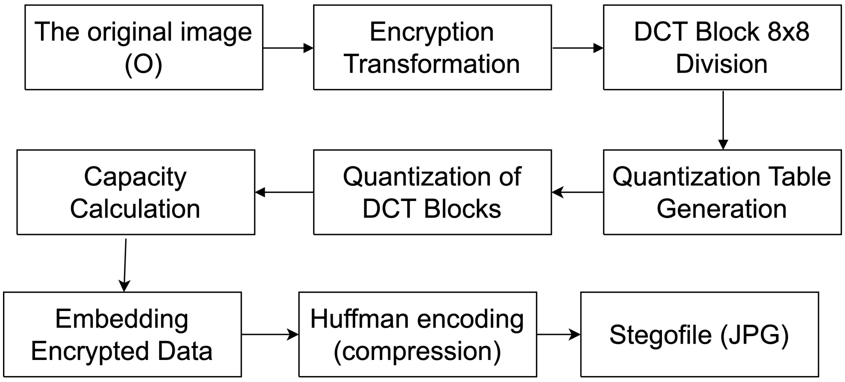

The original image (O)
Encryption Transformation
DCT Block 8x8 Division
Quantization Table Generation
Quantization of DCT Blocks
Capacity Calculation
Embedding Encrypted Data
Huffman encoding (compression)
Stegofile (JPG)

Figure 1. The proposed F5A steganographic method's encoding scheme.

- Specification of the initial data required:
  - The desired compression ratio (semi-secret component);
  - The data to be encoded (original image (O));
  - The encoding key—the name of the website where the image will be hosted.
- The carrier-image data (RGB pixels) are read.
- The scanned data are divided into non-overlapping blocks of 8 × 8 and a discrete cosine transform (DCT) is used to split the blocks into coefficient matrices.
- A quantization coefficient matrix is generated based on the given compression ratio.
- Quantization is performed on the basis of the generated quantization coefficient matrix.
- The available capacity is calculated and compared with the capacity of the secret data (the original image).
- The data of the original image are encrypted with a key.
- A cryptographically strong random number generator is generated from the given key.
- A permutation is performed (allocating the locations where the sensitive information will be stored according to the random number generator).
- The secret information is written to the allocated locations (encrypted original image).
- Hofmann encoding is performed, and the image is compressed.
- A JPG stegofile is obtained.

In DCT, a large proportion of the coefficients remain with 0 values, and the F5 algorithm does not use 0 values to encode secret information, so extending F5A to non-zero values using a modified quantization table increases the steganographic method's capacity.

The decoding process is also different from the normal operation of F5. The steps for F5A decoding are as follows (Figure 2):

- Specification of the data required for the encoding:
  - A stegofile in JPG format;
  - The decoding key—the name of the website where the image will be hosted.
- Huffman decoding is executed.
- The key is used to find the locations where sensitive information is embedded and decode it using the available key.
- The original image is obtained.

Further steps can be performed to extract the raw image of the carrier: de-quantization, inverse DCT function, and 8 × 8 block merging. However, it is advisable to make this step more difficult since it is the reconstruction of the exact carrier image that would reduce the eligibility of the proposed method for copyright protection. This step is already complicated by the fact that in order to properly decode the carrier image, it is necessary to know what compression ratio was used, so this component must remain partly secret.

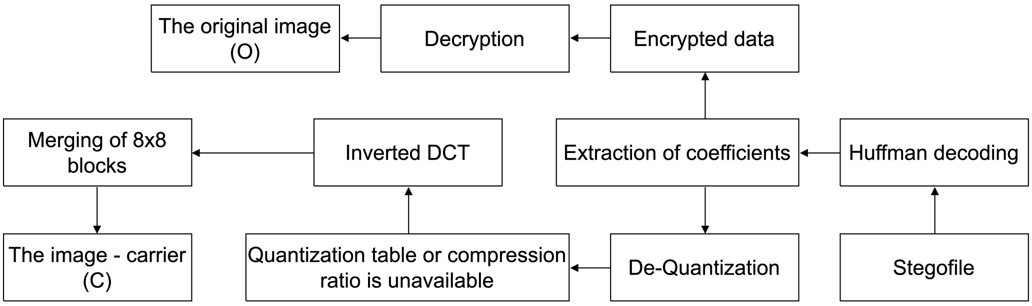


Figure 2. The proposed F5A steganographic method's decoding scheme.

By integrating the F5A method into the browser, many users will only be able to see content that has not been obtained through copyright infringement. It would check the incoming image, and if it is not allowed on the page or does not match the author's domain, the content can be masked, altered, or blocked at will. The method's steganographic function allows for the image to be hidden within another image. The idea is to allow the user to upload the original document and then protect it by masking, replacing it with another, or blocking it. The process of steganographic hiding also involves the inclusion of a key, which is essential for viewing the image. The processed image is then returned to the user, along with the key that reveals the real image.

In the F5A method, the image is encoded in such a way that the real image cannot be extracted from it, but it is not visually apparent to the user. The initial data parameters are as follows:

- F is the data known to all to confirm that the steganographic method is used.
- O is the original image that needs protection from unauthorized use.
- C stands for carrier, which refers to any image that is not relevant.
- O(F) is the validation data file embedded in the original image.
- C(O(F)) is the original image embedded in the carrier, encoding the validation data file.
- A is a normal, unprocessed image.

Using the new F5A method, a validation data file is steganographically inserted into the original image O. The embedding uses a key (verifyKEY), which is known to all. The result is O(F) (Figure 3). The O(F) obtained is then inserted into the carrier file C in the same way using F5A, this time using the domain name as the key. After inserting O(F) into C, the result is C(O(F)).

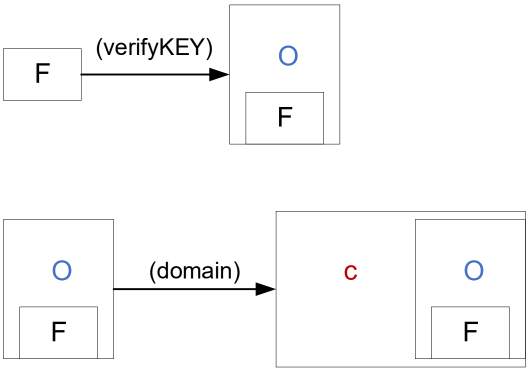


Figure 3. Image encoding workflow.

Checking whether the image is regular or processed using the F5A method is necessary during the decoding process to determine its displayability on the website. This is performed by first checking that the image is encoded according to the C(O(F)) scheme (Figure 4), that is, whether it can be decoded using the domain name of the website (first decoding test) and the verification key (verifyKEY) (second decoding test). If so, it will be decoded from C(O(F)) to O(F), and the original image will be presented to the user with the validation data token inserted. Visually, there is no difference from the original image.

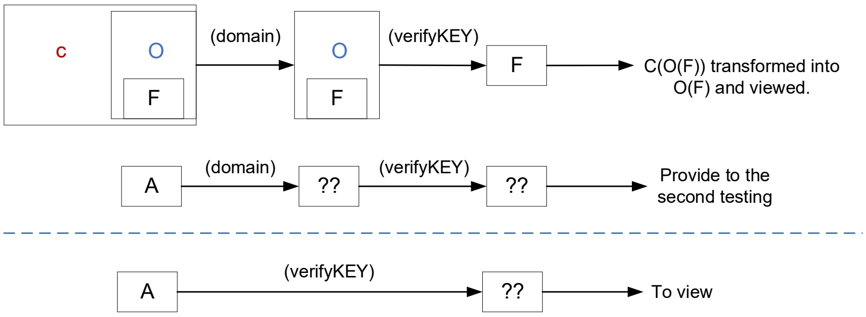

c
O
F
(domain)
O
F
(verifyKEY)
F
C(O(F)) transformed into O(F) and viewed.
A
(domain)
??
(verifyKEY)
??
Provide to the second testing
A
(verifyKEY)
??
To view

Figure 4. Image decoding workflow.

If content thieves tried to download the image, they would receive C(O(F)). And if an already decoded O(F) image is visible, it is possible to take a screenshot to obtain O(F). However, these cases have been thought about, and safeguards have been put in place. It is crucial to consider standard website images, identify and permit their display, and prevent the unauthorized display of O(F). If the first test fails, a second one is necessary to verify if the file has undergone processing using the F5A method. It is attempted to decode with verifyKEY, and if this succeeds, it can be concluded that the content is stolen. The "??" in the middle and bottom sections of the diagram indicates processes or transformations that are either unknown, or the first check that the file is processed by the F5A algorithm failed or not specified in the given context. However, if decoding with verifyKEY fails, the image is normal, and its display is allowed (Figure 5).

Copyrighted content can also be used for its intended purpose. By entering into a contract, or informal agreement, with the owner of the other domain for the use of the content, the F5A approach would allow third parties to use the content. The creation of a whitelisting function would allow the author to indicate which domains have consent to use the content and not be subject to the usual restrictions on these domains.

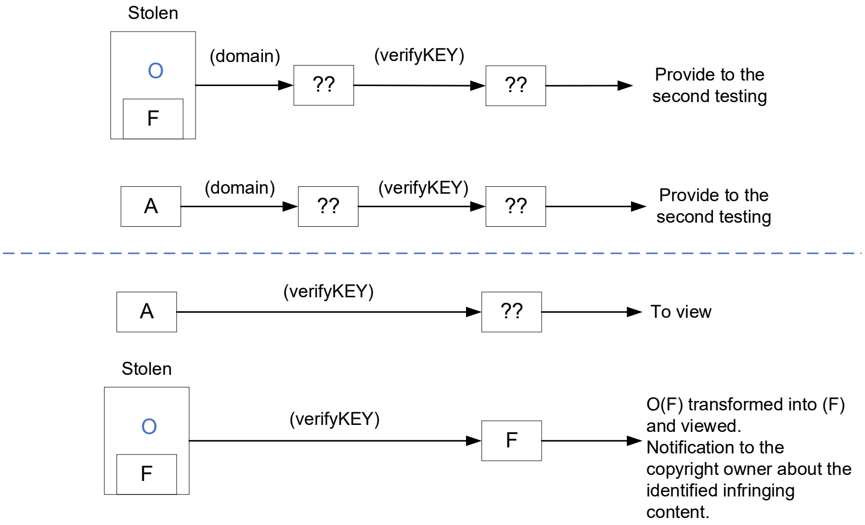

Stolen
O
F
(domain)
??
(verifyKEY)
??
Provide to the second testing
A
(domain)
??
(verifyKEY)
??
Provide to the second testing
A
(verifyKEY)
??
To view
Stolen
O
F
(verifyKEY)
F
O(F) transformed into (F) and viewed.
Notification to the copyright owner about the identified infringing content.

Figure 5. Workflow for the identification of the stolen image.

Often, copyrighted content is of particular importance to a company and can cause a lot of financial damage if it falls into the wrong hands. Consider a scenario where a company possesses confidential drawings that must remain within the company's boundaries. In such a case, there is a need to track the path of this content and identify exactly who leaked the information. The leak protection function of the F5A method will allow us to indicate to whom the copyright file was sent and, in the event of a security breach, pinpoint the offender.

4. Experimental Evaluation

In this section, the experimental setup and evaluation results are presented that demonstrate the efficacy of the F5A method compared to the traditional F5 algorithm. It covers empirical experiments and results, detailing the methodology, tools, and metrics used to assess the performance of the steganographic enhancements. The section aims to validate the improvements claimed by the F5A method through rigorous testing and analysis.

4.1. Empirical Experiments

4.1.1. Implementation of F5A Steganographic Method as the StegoGuard Tool

The StegoGuard plugin, a copyright protection tool based on the F5A method's principles, implemented the proposed F5A steganographic method. The main requirements for implementing the proposed method were as follows:

1. The selected compression ratio specifies the size of compression to be applied for data embedding. It is defined as a ratio and referred to as a semi-secret component, as it is not completely public yet not entirely secret.
2. The encoded data contain a specific image that will be concealed through the steganography process.
3. An encoding key is any phrase or word used as a secret key necessary for the decryption of the data. This adds an additional layer of security.
4. The carrier is the image into which the data to be encoded will be embedded. It acts as a data repository.

These requirements define the function of the F5A method, which protects digital information by integrating it into other media, ensuring data secrecy and copyright protection. Table 1 provides the initial JPEG data that were used for the experimental research.

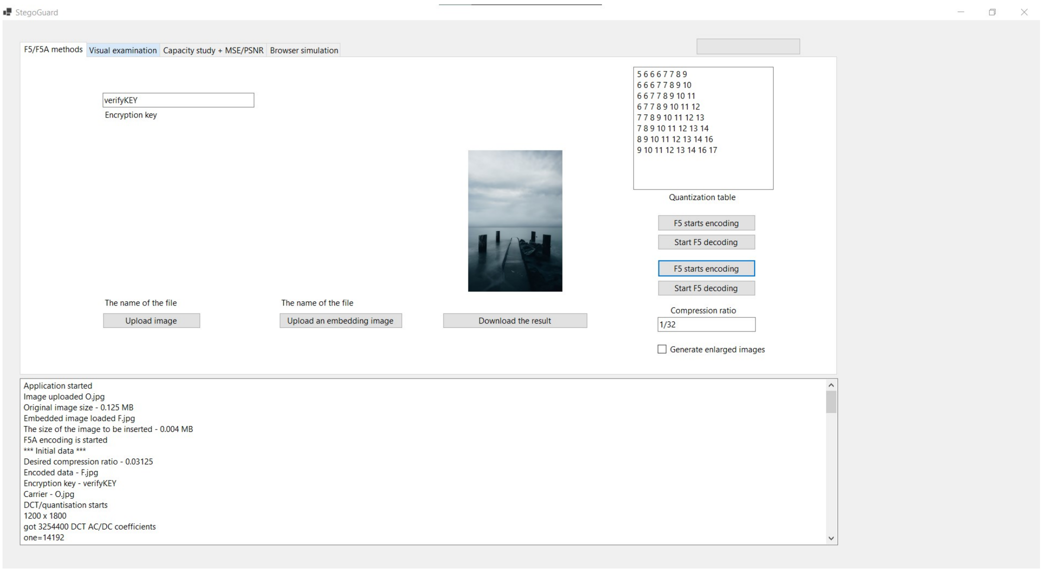
Firstly, the carrier data were read. The file in C# was read because a StegoGuard tool will be used as a browser plugin. The reading of data was performed from the uploaded file into a bit array structure. This structure exists in various languages, but in C# it is available in the language file's I/O API. The bit array type object was obtained from the scanned file using an implementation of the F5 algorithm. In the same way, it can read the actual image's data. After that, the scanned data (RGB pixel values) were divided into 8 × 8 blocks (Figure 6).

Table 1. Initial data for the experimental research.

Elements in the Method	File Size, MB
A	0.469
F	0.004
O	0.125
C	4.552
O(F) for F5	0.09
O(F) for F5A (1–32)	0.142
O(F) for F5A (8)	0.08

After splitting the data into blocks, the transformation of these blocks using the DCT was performed. This process mirrors the JPEG compression algorithm and Andreas

Westfeld's F5 algorithm. This transformation produces blocks measuring 8 × 8 with pixel frequency coefficients.



Encryption key: verifyKEY

Quantization table:

```
5 6 6 6 7 7 8 9
6 6 6 7 7 8 9 10
6 6 7 7 8 9 10 11
6 7 7 8 9 10 11 12
7 7 8 9 10 11 12 13
7 8 9 10 11 12 13 14
8 9 10 11 12 13 14 16
9 10 11 12 13 14 16 17
```

The name of the file

The name of the file

Compression ratio: 1/32

```
Application started
Image uploaded O.jpg
Original image size - 0.125 MB
Embedded image loaded F.jpg
The size of the image to be inserted - 0.004 MB
F5A encoding is started
*** Initial data ***
Desired compression ratio - 0.03125
Encoded data - F.jpg
Encryption key - verifyKEY
Carrier - O.jpg
DCT/quantisation starts
1200 x 1800
got 3254400 DCT AC/DC coefficients
one=14192
```

Figure 6. The concept of the StegoGuard tool.

Given a selected original image [35] and F5 encoding, the implemented StegoGuard tool outputs the following parameters:

```
Initial data:
 Desired compression ratio - 0.03125
 Encoded data - F.jpg
 Encryption key - verifyKEY
 Carrier - O.jpg
```

4.1.2. Generation of Optimized Quantization Tables

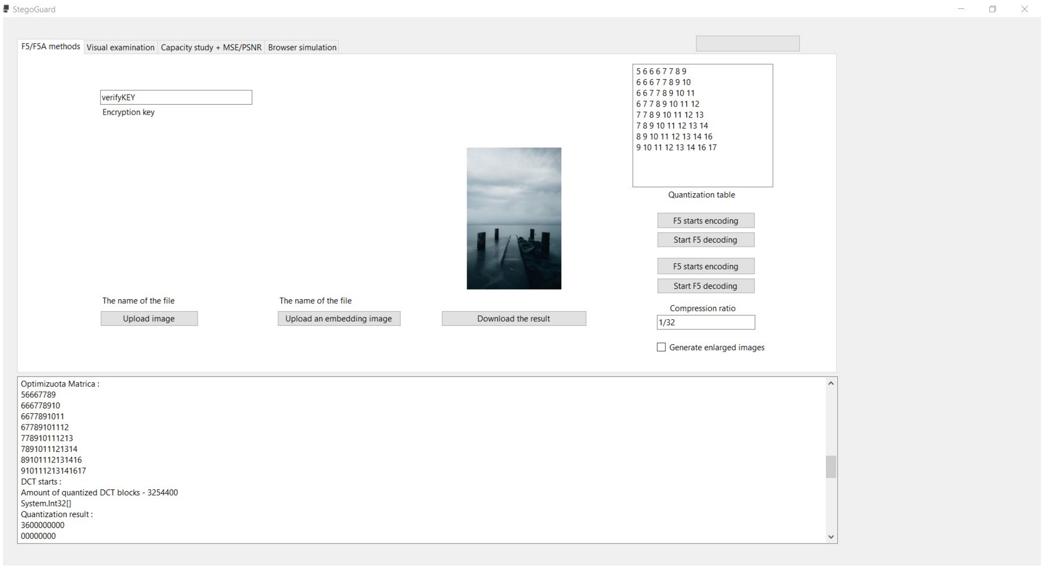
The generation of optimized quantization tables is a distinct feature of the F5A method. This process is undertaken to enhance the capacity of the existing F5 method and to introduce an additional feature that complicates the extraction of the carrier from the stegofile without knowledge of the compression ratio (Figure 7).

The F5 method uses standard JPEG quantization tables (Table 2), which maintain image quality according to the standard use of JPEG compression [36].

Table 2. Standardquantization table.

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Subsequently, the optimized quantization table is calculated according to the following Equation (1) [37] specified below, utilizing the desired compression ratio. This step involves applying the specified compression ratio. This equation shows that the optimal quantization table for the desired compression ratio (C_R) can be constructed by calculating the values of the coefficients A, D, and F.



Encryption key: verifyKEY

Quantization table:

```
5 6 6 6 7 7 8 9
6 6 6 7 7 8 9 10
6 6 7 7 8 9 10 11
6 7 7 8 9 10 11 12
7 7 8 9 10 11 12 13
7 8 9 10 11 12 13 14
8 9 10 11 12 13 14 16
9 10 11 12 13 14 16 17
```

Compression ratio: 1/32

□ Generate enlarged images

```
Optimizuota Matrica :
56667789
666778910
6677891011
67789101112
778910111213
7891011121314
89101112131416
910111213141617
DCT starts :
Amount of quantized DCT blocks - 3254400
System.Int32[]
Quantization result :
3600000000
00000000
```

Figure 7. Generation of optimised quantization table.

The resulting values of the coefficients A, D, F should then be substituted for each coefficient $Q_{x,y}$, using the Equation (2) [37].

$$\begin{cases} A = 5.43 + 2.15C_R \\ D = 0.0969 - 0.0565C_R + 0.00749C_R^2 \\ F = 1.83 \end{cases} \tag{1}$$

In this equation, x and y are the coordinates of the coefficients of the quantization table, and $z = x + y$ is the Manhattan distance from coordinate (0, 0).

$$Q_{x,y} = A + D_z{}^F \tag{2}$$

The output of the generated optimized quantization table using a StegoGuard tool, which was specially programmed by the author for this experiment. is provided in Table 3.

Table 3. Optimizedquantization table.

5	6	6	6	7	7	8	9
6	6	6	7	7	8	9	10
6	6	7	7	8	9	10	11
6	7	7	8	9	10	11	12
7	7	8	9	10	11	12	13
7	8	9	10	11	12	13	14
8	9	10	11	12	13	14	16
9	10	11	12	13	14	16	17

The step for the generation of the quantization tables is critical as it directly impacts the efficiency and security of the steganographic process by adjusting the granularity of the quantization, thereby affecting the embedded data's perceptibility and resilience against extraction attempts.

4.1.3. Quantization

Quantization is a compression method applied following the DCT, where the resulting matrix of coefficients is divided by the values in a quantization table on an element-wise basis. To perform this operation, one can utilize the existing implementation of the F5 algorithm. It is crucial that during the quantization process, there is an option to specify a custom quantization table, as using a standard table is unsuitable for the implementation of the F5A method. Following the quantization step, the resulting matrices typically display the majority of significant values clustered in the upper left corner, while many other coefficients are reduced to zero (see the output below). This distribution is key to maximizing data hiding effectiveness while minimizing the perceptual impact on the carrier image.

```
Quantization result :
3600000000
00000000
00000000
00000000
00000000
00000000
00000000
00000000
```

4.1.4. Matrix Encoding

Before proceeding with further steps, it is essential to perform a calculation of the potential capacity within the quantized matrices. The purpose of this step is to determine whether it is possible to embed the original file into the carrier. The existing implementation of the F5 algorithm in the C# language can execute this operation by calculating the number of bytes and comparing it with the number of possible encoded values in the quantized matrices. The length of the message to be embedded (in bytes) will be denoted as K. This makes sure that the embedding process can work without using up too much of the quantized matrix's storage space. This maintains the steganographic embedding's integrity and effectiveness.

Prior to performing the permutation—the distribution of encrypted image values across quantized matrix positions—a cryptographically strong random number generator is generated from the encoding key. The permutation is then executed with these randomly generated numbers and the count of coefficients (including zeros). During the permutation, the identification of locations within the matrix is performed in order to determine where it is needed to insert encrypted image values.

Ultimately, the quantized matrices embed the encrypted image information. Before starting the encoding, the calculation of the length of the buffer (BUFF), which will be involved in the encoding process, has to be performed (Equation (3)).

$$N = 2K - 1 \tag{3}$$

The embedding is executed in the following manner:

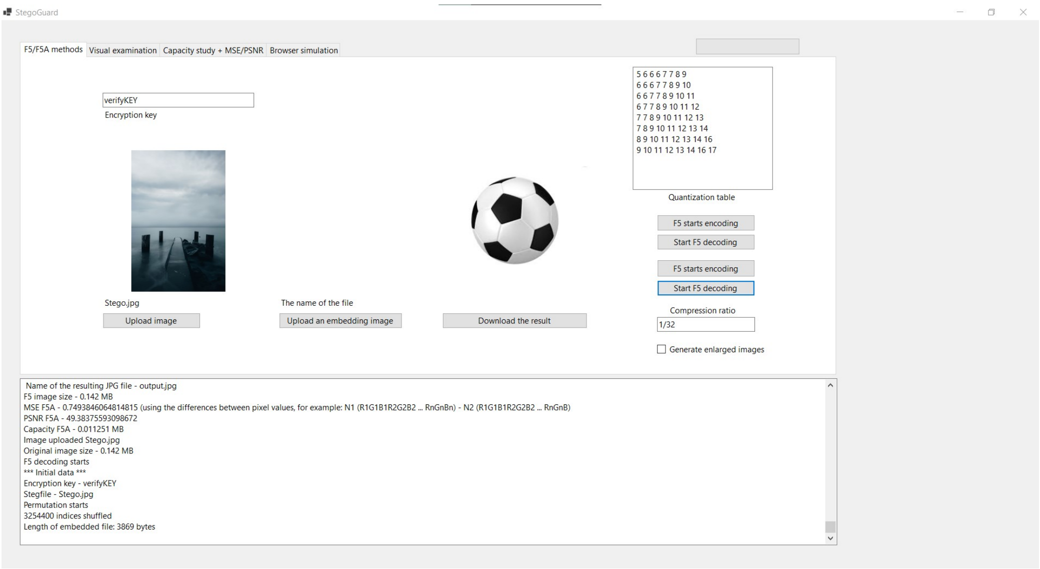
- BUFF filled with N non-zero coefficients from the coefficient matrix.
- XOR type hash function performed on the BUFF array. After the hash function, we should obtain a value, HASH, with a length of K (occupying that many bit positions).
- Sequentially K quantities of encrypted file values added to HASH in order to obtain the value SUM.
- If adding the bit results in SUM = 0, BUFF is left unchanged; otherwise, the values of SUM need to be assigned to the BUFF array. SUM[0...end] arranged into BUFF[0...N].
- Zero values are checked in the BUFF array. If present, zeros will be replaced with further values from non-zero coefficients (repeated from first step). If there are no zero values, then a new quantity of non-zero coefficients are taken and repeated from step 1 until there is no more information to encode.

The matrix encoding produces a complete buffer of values. This process will utilize the existing implementations of the F5 algorithm.

The final step is Huffman encoding. This step allows us to convert the buffer filled with matrix values into binary code, from which we retrieve our file data in JPEG format. This step was conducted using existing tools for the F5 algorithm implementation. After Huffman encoding, the final JPEG stegofile was produced.

4.1.5. F5A Decoding

Huffman decoding can also be used to reproduce the previous values of the buffer, from which the original information in the decoding algorithm can be reconstructed. This step was conducted using the existing tools of the F5 algorithm implementation. The Huffman decoding requires the stegofile produced by the F5A method (Figure 8).



Encryption key: verifyKEY

Quantization table:

```
5 6 6 6 7 7 8 9
6 6 6 7 7 8 9 10
6 6 7 7 8 9 10 11
6 7 7 8 9 10 11 12
7 7 8 9 10 11 12 13
7 8 9 10 11 12 13 14
8 9 10 11 12 13 14 16
9 10 11 12 13 14 16 17
```

Compression ratio: 1/32

```
 Name of the resulting JPG file - output.jpg
F5 image size - 0.142 MB
MSE F5A - 0.7493846064814815 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 49.38375593098672
Capacity F5A - 0.011251 MB
Image uploaded Stego.jpg
Original image size - 0.142 MB
F5 decoding starts
*** Initial data ***
Encryption key - verifyKEY
Stegfile - Stego.jpg
Permutation starts
3254400 indices shuffled
Length of embedded file: 3869 bytes
```

Figure 8. Results after F5A decoding.

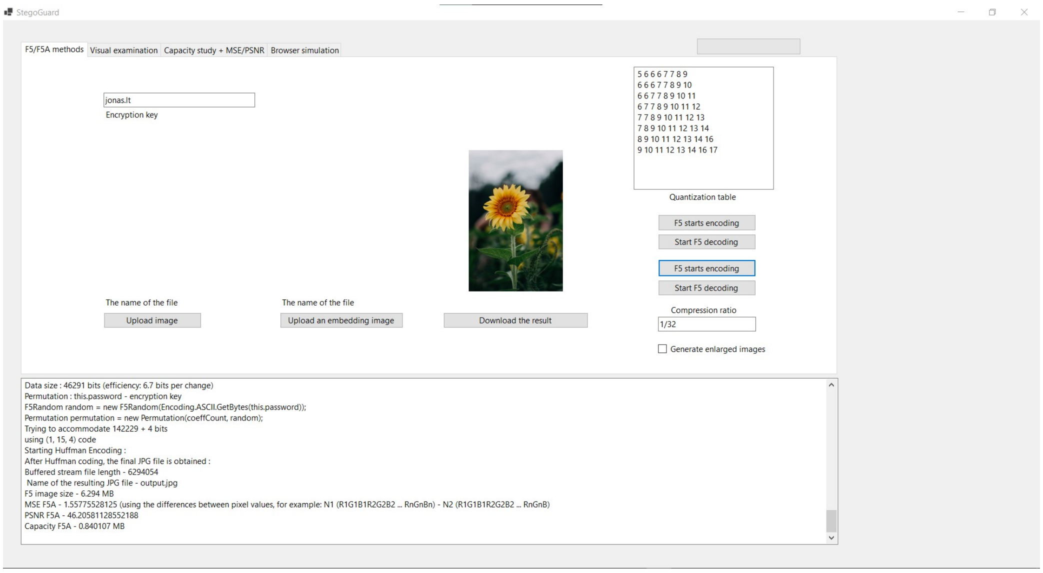
In order to extract the hidden information, it is necessary to know the encoding key (see below). Based on this key, a pseudo-random number generator is regenerated, and the positions within the file where information is encoded are determined.

```
Initial data:
 Encryption key - verifyKEY
 Stegfile - Stego.jpg
 Permutation starts
 3254400 indices shuffled
 Length of embedded file: 3869 bytes
```

Then, information is extracted from these positions, resulting in the recovery of the actual image. This process ensures that only individuals with access to the correct encoding key can retrieve the concealed information, maintaining the confidentiality and integrity of the data embedded within the stegofile.

4.1.6. Testing Scenario for StegoGuard Usage for Copyright Protection

The author encrypts the publicly known image F with the universally known key verifyKEY to produce O(F) (Figure 9). Then, O(F)—the original image with embedded verification data F—is embedded into the carrier file C. The domain name where the image will be displayed is used as the key.



Encryption key: jonas.lt

Quantization table:

```
5 6 6 6 7 7 8 9
6 6 6 7 7 8 9 10
6 6 7 7 8 9 10 11
6 7 7 8 9 10 11 12
7 7 8 9 10 11 12 13
7 8 9 10 11 12 13 14
8 9 10 11 12 13 14 16
9 10 11 12 13 14 16 17
```

Compression ratio: 1/32

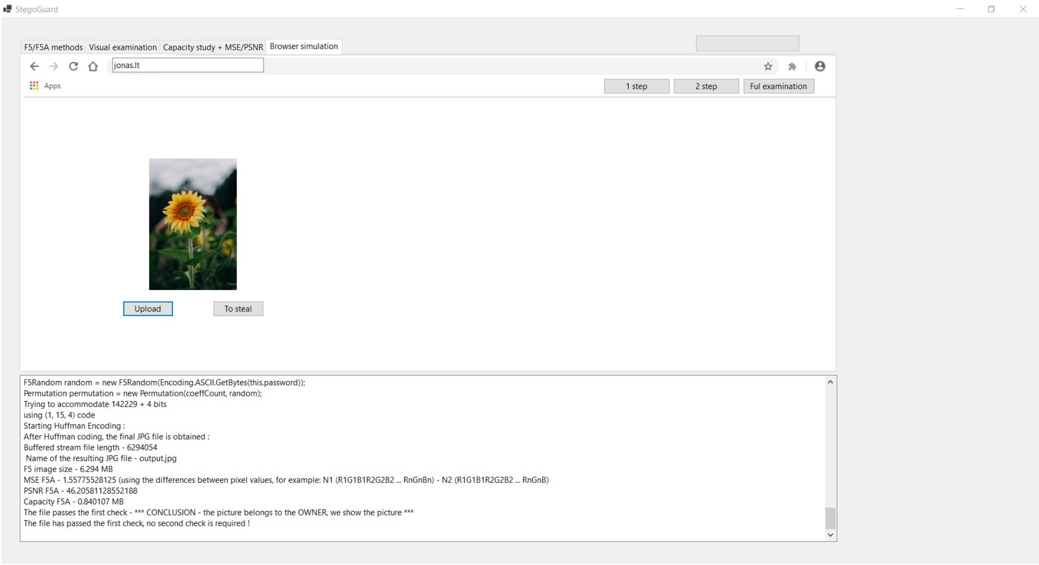
```
Data size : 46291 bits (efficiency: 6.7 bits per change)
Permutation : this.password - encryption key
F5Random random = new F5Random(Encoding.ASCII.GetBytes(this.password));
Permutation permutation = new Permutation(coeffCount, random);
Trying to accommodate 142229 + 4 bits
using (1, 15, 4) code
Starting Huffman Encoding :
After Huffman coding, the final JPG file is obtained :
Buffered stream file length - 6294054
 Name of the resulting JPG file - output.jpg
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
```

Figure 9. Producing of O(F).

Scenario for legally used content. The StegoGuard operation is based on two checks—attempts to decrypt and conclusions drawn from the results. Below is the part of the output after Huffman encoding:

```
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values,
for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
```

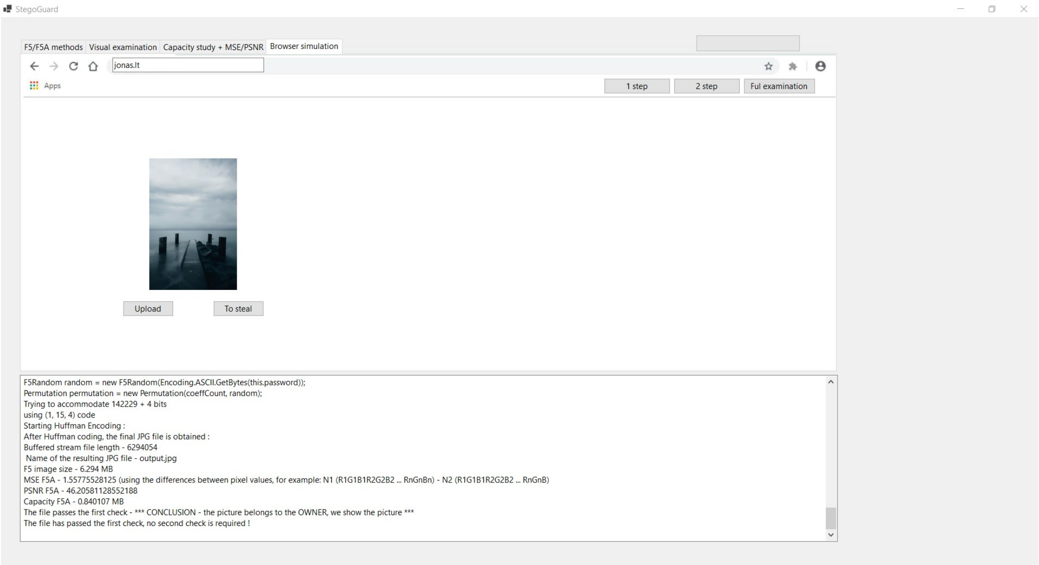
Checking if decryption can be performed based on the domain and then by the key verifyKEY can be performed by retrieving the publicly known image F (Figure 10). If successful, the content is confirmed to be legitimate (Figure 11).



```
F5Random random = new F5Random(Encoding.ASCII.GetBytes(this.password));
Permutation permutation = new Permutation(coeffCount, random);
Trying to accommodate 142229 + 4 bits
using (1, 15, 4) code
Starting Huffman Encoding :
After Huffman coding, the final JPG file is obtained :
Buffered stream file length - 6294054
 Name of the resulting JPG file - output.jpg
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
```

Figure 10. Decryption process.

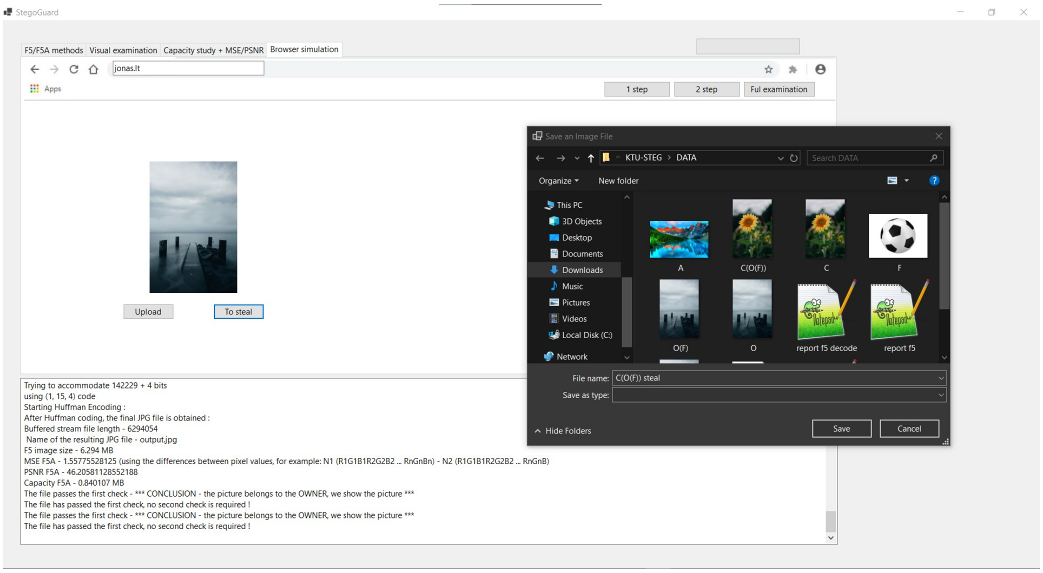
```
The file passes the first check - *** CONCLUSION - the picture belongs to
the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
```



```
F5Random random = new F5Random(Encoding.ASCII.GetBytes(this.password));
Permutation permutation = new Permutation(coeffCount, random);
Trying to accommodate 142229 + 4 bits
using (1, 15, 4) code
Starting Huffman Encoding :
After Huffman coding, the final JPG file is obtained :
Buffered stream file length - 6294054
 Name of the resulting JPG file - output.jpg
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
```

Figure 11. Result of the decryption.

During testing, it was observed that the first check is usually sufficient. The second decryption is not necessary to ensure the image belongs to the owner. If content decrypts with both keys, it is displayed; i.e., O(F) is shown (decrypted by domain) (Figure 12). This image is recognized as stolen and will be used in another scenario.

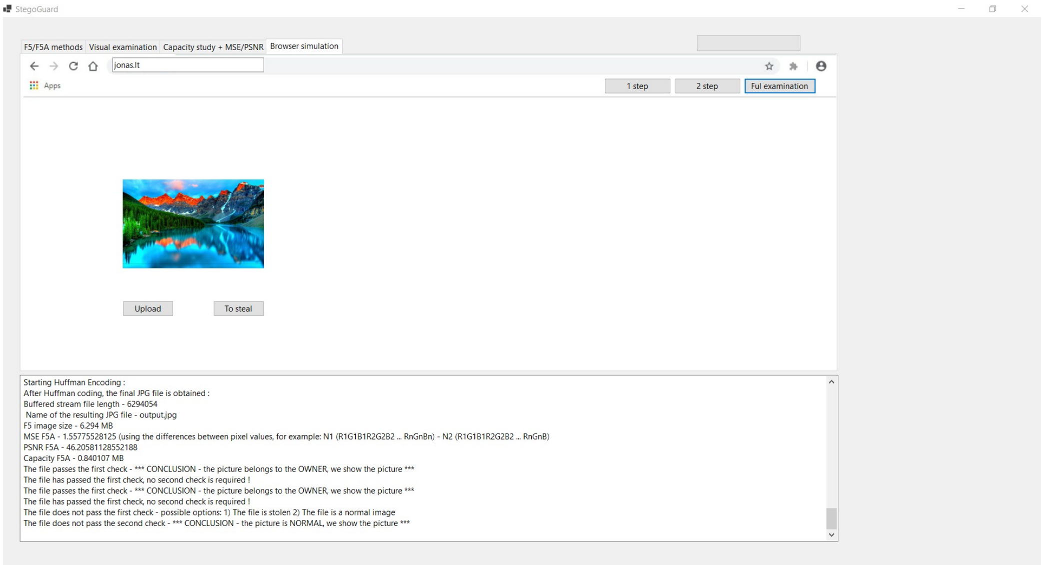


File name: C(O(F)) steal
Save as type:

```
Trying to accommodate 142229 + 4 bits
using (1, 15, 4) code
Starting Huffman Encoding :
After Huffman coding, the final JPG file is obtained :
Buffered stream file length - 6294054
 Name of the resulting JPG file - output.jpg
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
```

Figure 12. Decryption by the domain.

It is also important that the user's website can display simple, unencrypted images, so we must also check the operation of A (Figure 13). It should not decrypt with either key—domain or verifyKEY.



```
Starting Huffman Encoding :
After Huffman coding, the final JPG file is obtained :
Buffered stream file length - 6294054
 Name of the resulting JPG file - output.jpg
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
The file does not pass the first check - possible options: 1) The file is stolen 2) The file is a normal image
The file does not pass the second check - *** CONCLUSION - the picture is NORMAL, we show the picture ***
```

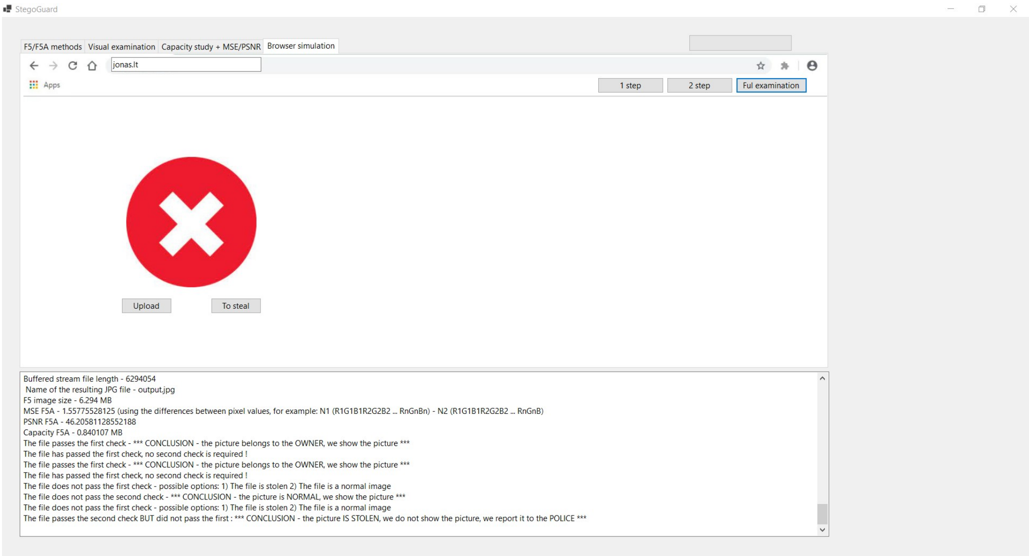
Figure 13. Cheking the operation of A.

The file does not decrypt with either key, so it is a standard unprocessed image and should be displayed on the website:

```
The file does not pass the first check - possible options:
(1) The file is stolen (2) The file is a normal image
The file does not pass the second check -
*** CONCLUSION - the picture is NORMAL, we show the picture ***
```

Scenario for stolen content (Figure 14). The StegoGuard operation is based on two checks:

- The operation of A.jpg file is similar to the above. An unprocessed image will be displayed. (It will not decrypt with either key.)
- The case of a stolen image placed on the thief "vagis.lt" website then can be examined.



```
Buffered stream file length - 6294054
 Name of the resulting JPG file - output.jpg
F5 image size - 6.294 MB
MSE F5A - 1.55775528125 (using the differences between pixel values, for example: N1 (R1G1B1R2G2B2 ... RnGnBn) - N2 (R1G1B1R2G2B2 ... RnGnB)
PSNR F5A - 46.20581128552188
Capacity F5A - 0.840107 MB
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
The file passes the first check - *** CONCLUSION - the picture belongs to the OWNER, we show the picture ***
The file has passed the first check, no second check is required !
The file does not pass the first check - possible options: 1) The file is stolen 2) The file is a normal image
The file does not pass the second check - *** CONCLUSION - the picture is NORMAL, we show the picture ***
The file does not pass the first check - possible options: 1) The file is stolen 2) The file is a normal image
The file passes the second check BUT did not pass the first : *** CONCLUSION - the picture IS STOLEN, we do not show the picture, we report it to the POLICE ***
```

Figure 14. Stolen image.

The crucial difference from an unprocessed image is that it decrypts with the publicly known key verifyKEY. This indicates that the image is indeed stolen:

```
The file does not pass the first check - possible options:
(1) The file is stolen (2) The file is a normal image
The file passes the second check BUT did not pass the first:
*** CONCLUSION - the picture IS STOLEN,
we do not show the picture, we report it to the POLICE ***
```

4.2. Experimental Results of F5A Method

The proposed steganographic method, F5A, has the following validation criteria: (a) capacity—the method must be able to hide as much information as possible. As the image will hide a full copy of the image, it must remain of the best quality, so the amount of data to be hidden is particularly important for this scenario and (b) to reliably protect the hidden content from disclosure, the method must easily combine with an embeddable key to encrypt the information. The goal of the research is to determine:

1. What is the impact of using different compression ratios in optimized quantization tables?
2. To what extent can the use of optimized quantization tables increase the steganographic method's capacity?

The developed F5A steganographic method will be validated and compared with the F5 steganographic algorithm by means of steganographic evaluation methods that meet the set criteria. Both solutions use an embeddable key, so the embeddable key criterion is not evaluated. The following are used to determine the impact of the use of optimized quantization tables: the process calculates the file size after steganographic encoding; calculation of the mean square error (MSE); calculation of the peak signal-to-noise ratio (PSNR); and calculation of the maximum amount of hidden data. Modifications to the image are made in order to determine the robustness of F5 and F5A algorithms. MSE and PSNR are fundamental metrics in image steganography that help evaluate the trade-offs between the amount of hidden data, the perceptibility of changes, and the overall security and efficiency of the steganographic method. Mathematically, MSE is defined as [38]

$$MSE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (O(i,j) - S(i,j))^2 \tag{4}$$

where $O(i,j)$ represents the pixel value at position (i,j) in the original image; $S(i,j)$ represents the pixel value at position (i,j) in the stenographic image; M and N are the dimensions of the image.

PSNRis defined as presented in Equation (5) [38], where PV is the maximum value of pixels in the image.

$$PSNR = 10log_{10}\left(\frac{PV^2}{MSE}\right) \tag{5}$$

Tables 4 and 5 provide the main results obtained during the validation.

During O(F) encoding, increasing the compression ratio decreases the size of the steganogram file. F5A—from compression ratio 16 onwards, no values are available because the encoding is no longer possible, and the embedded file no longer fits. With a compression ratio of 8, the steganographic algorithm could be optimized to obtain the smallest possible steganogram file size, thus improving the F5 algorithm. The F5A algorithm, with a compression ratio of 1/128, increases the steganogram file size by a factor of 1.577 in the O(F) case compared to the F5 algorithm (the F5 algorithm results in a file size of 4.075 MB, an MSE of 4.9168, a PSNR of 41.24, and a storage capacity of 0.5458 MB).

Table 4. O(F) data obtained during the experimental research.

Compresion Ratio	File Size, MB	MSE	PSNR	Capacity, MB
0	0.142	0.7496	49.3824	0.0113
1/128	0.142	0.7496	49.3824	0.0113
1/64	0.142	0.7496	49.3824	0.0113
1/32	0.142	0.7496	49.3824	0.0113
1/16	0.142	0.7496	49.3824	0.0113
1/8	0.142	0.7496	49.3824	0.0113
1/4	0.138	0.7564	49.3432	0.0110
1/2	0.131	0.7834	49.1908	0.0107
1	0.129	0.7851	49.1817	0.0106
2	0.108	0.9288	48.4516	0.008
4	0.095	1.0707	47.8341	0.0071
8	0.08	1.6272	46.0165	0.0057

As the compression ratio increases during O(F) coding, the MSE value increases. An increasing MSE value may indicate that more information is being hidden or may indicate an inefficiency of the method or a greater discrepancy with the original image. Given that the study consistently hid the same amount of information in all cases, we can conclude that an increase in the compression ratio leads to an increase in the mean squared error (MSE), thereby reducing the efficiency of the method. However, the performance of the F5 method using the standard JPG quantization table is lower than that of the F5A method with a compression ratio lower than 4. At a compression ratio of 1/128, the F5A method successfully increases the steganogram efficiency, and in the case of O(F), we observe a 1.716-fold increase in the steganogram efficiency in terms of the MSE value compared to the F5 algorithm.

A higher PSNR value typically indicates a better quality of the file processed by steganographic methods. For a typical JPG file, PSNR values between 30 and 50 dB are common, and values above 40 dB indicate very good image quality. Therefore, although there is a visible difference between the F5 and F5A methods, they both fall within the range of very good image quality. The F5A successfully enhances image quality at a compression ratio of 1/128, achieving a 1.121-fold increase in PSNR value for O(F) compared to the F5 algorithm.

The F5A method, with a compression ratio of 1/128, can successfully increase the capacity of the steganographic algorithm, and in the case of O(F), we observe a 1.693-fold increase in the method's capacity compared to the F5 algorithm.

Table 5. C(O(F)) data obtained during the experimental research.

Compresion Ratio	File Size, MB	MSE	PSNR	Capacity, MB
0	6.294	1.5579	46.2054	0.8401
1/128	6.294	1.5579	46.2054	0.8401
1/64	6.294	1.5579	46.2054	0.8401
1/32	6.294	1.5579	46.2054	0.8401
1/16	6.294	1.5579	46.2054	0.8401
1/8	6.294	1.5579	46.2054	0.8401
1/4	6.116	1.5935	46.1074	0.8243
1/2	5.903	1.6113	46.0589	0.8079
1	5.855	1.6554	45.9419	0.8078
2	5.391	1.8521	45.4542	0.7535
4	4.562	2.6350	43.923	0.608
8	3.633	5.5051	40.7232	0.4407
16	1.545	18.9508	35.3545	0.1585

C(O(F)) at encoding time F5A—from compression ratio 32, no values are available because the encoding is no longer possible and the inserted file no longer fits. When encoding a larger image size, C(O(F)) can be encoded with a compression ratio of 16, but this compression ratio, although it is able to reduce the final size of the steganogram file to a very significant extent, also makes the steganogram inefficient, with a large discrepancy between the steganogram and the original image. At a compression ratio of 1/128, the F5A method successfully increases the steganogram's efficiency, and for C(O(F)), we observe a 3.156-fold increase in the steganogram's efficiency in terms of the MSE compared to the F5 algorithm. (The F5 algorithm results in a file size of 4.075 MB, an MSE of 4.9168, a PSNR of 41.214, and a storage capacity of 0.5458 MB.)

When encoding a larger image size, C(O(F)) can be encoded with a compression ratio of 16. Similarly, when encoding the larger image size C(O(F)), the PSNR values for the F5 and F5A methods are relatively lower, and there is also a larger gap between the F5 and F5A methods. When the compression ratio reaches a value of 16, the quality of the image is no longer classified as very good quality, as the value is not higher than 40 dB.

When encoding a larger image size, C(O(F)) can be encoded with a compression ratio of 16. However, this greatly reduces the capacity of the steganographic algorithm. In the case of encoding a larger image size with C(O(F)), a capacity increase is also observed. It can be seen that the F5A algorithm, at a compression ratio of 1/128, is also able to successfully increase the capacity of the steganographic algorithm, with an increase of 1.539 times the capacity of the steganographic algorithm in the case of the C(O(F)) algorithm compared to the F5 algorithm.

The security aspect is evident through the method's performance across different metrics, particularly MSE and PSNR, which are crucial for maintaining the secrecy and integrity of the embedded data. Lower MSE and higher PSNR at strategic compression levels suggest that the method is robust against potential steganalysis.

5. Discussion

This study's findings underscore the robustness and versatility of the F5A steganographic method for enhancing digital copyright protection. The integration of Huffman coding within the F5A framework notably advances the security and efficiency of data embedding processes. By leveraging Huffman's ability to minimize file size without compromising data integrity, the F5A method facilitates the embedding of larger amounts of data into digital images, thus significantly reducing the risk of detection.

The dual-level encryption mechanism—first using a domain-specific key and secondarily through the widely recognized "verifyKEY"—provides a comprehensive security layer that deters unauthorized access and extraction of embedded data. Our tests confirmed that content is only displayed or accessible when both keys validate the decryption process, ensuring that the data remains secure against unauthorized decryption attempts.

Moreover, the application of the F5A method within a browser plugin context has demonstrated practical viability and effectiveness. The method's capability to adapt to different operational scenarios—whether the content is legally used or potentially stolen—highlights its potential for real-world applications in safeguarding digital copyrights. The scenario-based validation, particularly the ability to differentiate between legitimately used and stolen content, reinforces the algorithm's utility in forensic and copyright enforcement contexts.

The challenges remain, particularly concerning the method's performance in environments with varying data sizes and types. Future research could explore the scalability of the F5A method, assessing its effectiveness across different media formats and larger data sets. Additionally, investigating the impact of more complex encryption keys and enhancing the method's resistance to sophisticated steganalysis techniques would further solidify its applicability in digital copyright protection.

The F5A method represents a significant step forward in digital steganography, offering a nuanced and powerful tool for copyright holders to protect their assets while maintaining control over their distribution and use in the digital domain. Further exploration and development of this technology will undoubtedly enrich the field of digital media security.

6. Conclusions and Future Work

This paper successfully demonstrated the F5A steganographic method's capabilities in providing an effective and secure method for protecting digital copyrights by embedding hidden data within digital media. Adding discrete cosine transform (DCT) and Huffman coding to the F5A framework not only protects the hidden data's integrity but also reduces the file size, which makes the embedding process faster and harder to spot. Testing revealed that the newly created F5A method, using different compression ratios, can significantly improve various steganogram parameters compared to the traditional F5 method. It was observed that the F5A method improves the MSE index by a factor of 1.716 and the PSNR index by a factor of 1.121 while increasing the steganographic capacity of the image by 1.693 times for a smaller image, O(F). For a larger image, an increase in MSE value by 3.156 times, improved PSNR by 1.121 times, and increased algorithm capacity by 1.539 times were noted. Additionally, changes in the size of the steganogram files were observed, with increases of 1.577 times in the O(F) case and 1.544 times in the C(O(F)) case.

Future directions for this research include expanding the method's applicability to different media types, such as audio and video, which may present new challenges and opportunities for steganographic techniques. Further refinement of the encryption mechanisms could enhance the algorithm's resistance to increasingly sophisticated steganography methods. The F5A method represents a promising advancement in the field of digital copyright protection, offering a powerful tool for copyright holders to discreetly safeguard their content. Continued innovation and research are essential to keeping pace with the evolving landscape of digital media and copyright infringement.

Author Contributions: Conceptualization, A.V. and J.P.; methodology, R.B. and A.V.; software, Š.G. and J.P.; validation, R.B., Š.G., and J.P.; formal analysis, J.P.; investigation, R.B., Š.G., and J.P.; resources, Š.G. and J.P.; data curation, J.P.; writing—original draft preparation, R.B. and Š.G.; writing—review and editing, A.V.; visualization, Š.G.; supervision, A.V.; project administration, A.V.; funding acquisition, A.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this paper are available on request from the corresponding author. The data are not publicly available due to the project not being completed.

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

References

1. Chen, Y.; Hu, X.; Xiao, F. Digital Media Copyright Protection Technology in the Age of All Media. In *Data Processing Techniques and Applications for Cyber-Physical Systems (DPTA 2019)*; Huang, C., Chan, Y.W., Yen, N., Eds.; Advances in Intelligent Systems and Computing; Springer Singapore: Singapore, 2020; Volume 1088, pp. 843–850. [CrossRef]
2. Rai, P. Copyright Laws and Digital Piracy in Music Industries: The Relevance of Traditional Copyright Laws in the Digital Age and How Music Industries Should Cope with the Ongoing Piracy Culture. Master's Thesis, University of Agder, Kristiansand, Norway, 2020.
3. Venugopal, A.V. Copyright concerns of digital images in social media. *J. World Intellect. Prop.* **2020**, *23*, 579–597. [CrossRef]
4. Stim, R. *Getting Permission: Using & Licensing Copyright-Protected Materials Online & Off*; Nolo: Berkeley, CA, USA, 2022.
5. Dobre, R.A.; Preda, R.O.; Badea, R.A.; Stanciu, M.; Brumaru, A. Blockchain-based image copyright protection system using JPEG resistant digital signature. In Proceedings of the 2020 IEEE 26th International Symposium for Design and Technology in Electronic Packaging (SIITME), Pitesti, Romania, 21–24 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 206–210.
6. Inshakova, A.O.; Deryugina, T.V.; Malikov, E.Y. Intellectual property exchange as a platform for exclusive copyright transfer by means of smart contracts. In Proceedings of the 13th International Scientific and Practical Conference-Artificial Intelligence Anthropogenic Nature vs. Social Origin, Volgograd, Russia, 19–20 March 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 693–705.
7. Frattolillo, F. Digital copyright protection: Focus on some relevant solutions. *Int. J. Commun. Netw. Inf. Secur.* **2017**, *9*, 282. [CrossRef]

8. Megías, D.; Kuribayashi, M.; Qureshi, A. Survey on decentralized fingerprinting solutions: Copyright protection through piracy tracing. *Computers* **2020**, *9*, 26. [CrossRef]
9. Kadian, P.; Arora, S.M.; Arora, N. Robust Digital Watermarking Techniques for Copyright Protection of Digital Data: A Survey. *Wirel. Pers. Commun.* **2021**, *118*, 3225–3249. [CrossRef]
10. Dhawan, S.; Gupta, R. Analysis of various data security techniques of steganography: A survey. *Inf. Secur. J. A Glob. Perspect.* **2021**, *30*, 63–87. [CrossRef]
11. Yerby, J.; Breese, J. Applied Steganography: An Interesting Case for Learners of all Ages. *Cybersecur. Pedagog. Pract. J.* **2023**, *2*, 17.
12. Muralidharan, T.; Cohen, A.; Cohen, A.; Nissim, N. The infinite race between steganography and steganalysis in images. *Signal Process.* **2022**, *201*, 108711. [CrossRef]
13. Evsutin, O.; Melman, A.; Meshcheryakov, R. Digital steganography and watermarking for digital images: A review of current research directions. *IEEE Access* **2020**, *8*, 166589–166611. [CrossRef]
14. Subramanian, N.; Elharrouss, O.; Al-Maadeed, S.; Bouridane, A. Image steganography: A review of the recent advances. *IEEE Access* **2021**, *9*, 23409–23423. [CrossRef]
15. Li, L.; Zhang, X.; Chen, K.; Feng, G.; Wu, D.; Zhang, W. Image Steganography and Style Transformation Based on Generative Adversarial Network. *Mathematics* **2024**, *12*, 615. [CrossRef]
16. Mansouri, H.; Tahiri, M.A.; Bencherqui, A.; Moustabchir, H.; Qjidaa, H.; Sayyouri, M. Securing Color Images with an Innovative Hybrid Method Combining DNA Computing and Chaotic Systems. *Stat. Optim. Inf. Comput.* **2024**, *12*, 697–712. [CrossRef]
17. Tahiri, M.A.; Karmouni, H.; Bencherqui, A.; Daoui, A.; Sayyouri, M.; Qjidaa, H.; Hosny, K.M. New color image encryption using hybrid optimization algorithm and Krawtchouk fractional transformations. *Vis. Comput.* **2023**, *39*, 6395–6420. [CrossRef]
18. Tahiri, M.A.; Bencherqui, A.; Karmouni, H.; Amakdouf, H.; Mirjalili, S.; Motahhir, S.; Abouhawwash, M.; Askar, S.; Sayyouri, M.; Qjidaa, H. Implementation of a steganography system based on hybrid square quaternion moment compression in iomt. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 101604. [CrossRef]
19. Naz, M.T.S.A.; Zade, S. A New Approach for Image Steganography Using Inter Pixel Value Difference and Quantized Range Table Method. *Int. J. Sci. Res. Eng. Trends* **2022**, *8*, 898–903.
20. Tiwari, K.; Gangurde, S.J. LSB steganography using pixel locator sequence with AES. In Proceedings of the 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), Jalandhar, India, 21–23 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 302–307.
21. Fateh, M.; Rezvani, M.; Irani, Y. A new method of coding for steganography based on LSB matching revisited. *Secur. Commun. Netw.* **2021**, *2021*, 6610678. [CrossRef]
22. Wu, D.C.; Shih, Z.N.; Wu, J.H. Modified multiway pixel-value differencing methods based on general quantization ranges for image steganography. *IEEE Access* **2021**, *10*, 8824–8839. [CrossRef]
23. Almaliki, A.J.Q.; Abd, S.M.; Lafta, I.A.; Din, R.; Ghazali, O.; Almaliki, J.Q.; Utama, S. Application of the Canny Filter in Digital Steganography. *Semarak Int. J. Appl. Sci. Eng. Technol.* **2024**, *1*, 36–45.
24. Dai, H.; Cheng, J.; Li, Y. A Novel Steganography Algorithm Based on Quantization Table Modification and Image Scrambling in DCT Domain. *Int. J. Pattern Recognit. Artif. Intell.* **2021**, *35*, 2154001. [CrossRef]
25. Aloraini, M.; Sharifzadeh, M.; Schonfeld, D. Quantized Gaussian JPEG steganography and pool steganalysis. *IEEE Access* **2022**, *10*, 38031–38044. [CrossRef]
26. Zeng, K.; Chen, K.; Zhang, W.; Wang, Y.; Yu, N. Robust steganography for high quality images. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 4893–4906. [CrossRef]
27. Kelesidis, E.A.; Maimuţ, D.; Ciocan, I.T. Searching for Gemstones: Flawed Stegosystems May Hide Promising Ideas. In Proceedings of the International Conference on Codes, Cryptology, and Information Security, Rabat, Morocco, 29–31 May 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 242–260.
28. Zhang, J.; Zhao, X.; He, X.; Zhang, H. Improving the robustness of JPEG steganography with robustness cost. *IEEE Signal Process. Lett.* **2021**, *29*, 164–168. [CrossRef]
29. Wang, J.; Yang, C.; Wang, P.; Song, X.; Lu, J. Payload location for JPEG image steganography based on co-frequency sub-image filtering. *Int. J. Distrib. Sens. Netw.* **2020**, *16*, 155014771989956. [CrossRef]
30. Giboulot, Q.; Cogranne, R.; Borghys, D.; Bas, P. Effects and solutions of cover-source mismatch in image steganalysis. *Signal Process. Image Commun.* **2020**, *86*, 115888. [CrossRef]
31. Zhu, L.; Luo, X.; Yang, C.; Zhang, Y.; Liu, F. Invariances of JPEG-quantized DCT coefficients and their application in robust image steganography. *Signal Process.* **2021**, *183*, 108015. [CrossRef]
32. Liu, J.; Yang, C.; Wang, J.; Shi, Y. Stego key recovery method for F5 steganography with matrix encoding. *EURASIP J. Image Video Process.* **2020**, *2020*, 40. [CrossRef]
33. Liu, J.; Wang, Y.; Yang, Z.; Zhang, R.; Zhang, R. A Controllable Image Steganography with Chaos and User Key. In Proceedings of the Image and Graphics: 11th International Conference, ICIG 2021, Haikou, China, 6–8 August 2021; Part I 11; Springer: Berlin/Heidelberg, Germany, 2021; pp. 785–797.
34. Tutuncu, K.; Demirci, B. Image Steganography Methods and Performance Comparison. In Proceedings of the 4th International Conference on Advanced Technology & Sciences (ICAT'Rome), Rome, Italy, 23–25 November 2016; ATScience Group: London, UK, 2016; pp. 149–153.

35. Verne, H. Fog on Dark Waters Edge Photo. 2023. Avaliable online https://www.shopify.com/stock-photos/photos/fog-on-dark-waters-edge?c=wallpapers (accessed on 12 April 2024).
36. Wang, Q.; Liu, P.; Zhang, L.; Cheng, F.; Qiu, J.; Zhang, X. Rate–distortion optimal evolutionary algorithm for JPEG quantization with multiple rates. *Knowl.-Based Syst.* **2022**, *244*, 108500. [CrossRef]
37. Hopkins, M.; Mitzenmacher, M.; Wagner-Carena, S. Simulated annealing for jpeg quantization. *arXiv* **2017**, arXiv:1709.00649.
38. Rustad, S.; Andono, P.N.; Shidik, G.F. Digital image steganography survey and investigation (goal, assessment, method, development, and dataset). *Signal Process.* **2023**, *206*, 108908.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.