Fingerprinted Secret Sharing Steganography for Robustness against Image Cropping Attacks

Vidyasagar M. Potdar, Song Han, Elizabeth Chang

School of Information Systems, Curtin University of Technology, Perth, Western Australia
e-mail: Vidyasagar.Potdar, Song.Han, Elizabeth.Chang@cbs.curtin.edu.au

Abstract — Steganography is the art and science of hiding information. In this paper we propose a conceptual framework for Fingerprinted Secret Sharing Steganography. We offer a technique to break the main secret into multiple parts and hide them individually in a cover medium. We use a novel technique to compress the data to a considerable extent. We use the Lagrange Interpolating Polynomial method to recover the shared secret. We also show how the proposed technique can offer robust mechanism to protect data loss because of image cropping. We use the (k,n) threshold scheme to decide the minimum number of parts required to recover the secret data completely. The security of our scheme is based on the security principle of steganography and secret sharing scheme.

Index Terms — Secret Sharing Scheme, Information Hiding, Data Compression, Steganography, Data Embedding Capacity.

I. INTRODUCTION

Cryptography is the art and science of scrambling information for covert transmission between predetermined individuals, groups or organizations. The phenomenon has an extensive and interesting history, and evidence for cryptographic use by Egyptian civilization 4000 years ago has beeen gathered by archaeologists. Given its prior use in secret communication, it has always generated the interest of excluded but interested parties. Unauthorized persons have at times been successful in deciphering covert messages; nations have even crumbled as a result.

A contemporary phenomenon is the information hacker, who searches for encrypted content across the network. In such a situation we need an alternate approach to protect secret information from being lost to unauthorized parties. Steganography provides a good alternative. Cryptographic techniques attempt to conceal the content of messages, whereas steganographic techniques conceal the very existence of the secret messages.

Steganography, meaning "covered writing", includes methods of transmitting secret message through innocuous looking cover mediums in such a manner that the existence of the hidden message is undetectable. Using steganographic techniques we can hide secret information in digital image files, digital audio and video files, or any other digital medium that has some redundant bits that can be replaced to hide secret data. Pre-computing steganography has a long history but digital steganography as a research field is avante garde. A simple steganographic technique is described in Fig. 1.
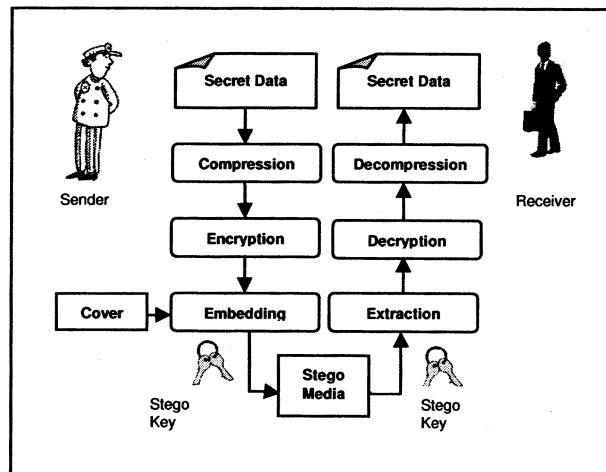

Fig.1 Generic Steganographic Technique

II. ADDRESSED PROBLEM

In this paper we address the issue of steganographic data loss because.Image cropping is a major issue in image steganography. Hence we propose a conceptual framework to protect this information loss. A traditional technique to tackle this problem is to hide data multiple times; however, this results in low embedding capacity. Thus we see that there is always a tradeoff between embedding capacity and robustness with respect to cropping.. In this paper we try to narrow down this margin. To do this we use the secret sharing scheme to hide the data in the image. We hide the data in multiple logical locations within the image. We use the Lagrange Interpolating Polynomial technique to retrieve the data from the image.

The proposed technology – *fingerprinted secret sharing steganography* – could be utilized in the following application scenario (in which some people are more important than others), which is related to logistics informatics.

Suppose there is a Logistics Management and Transport company, *WEST FIELD*, which has to securely

store a confidential document (this document will determine the future of *WEST FIELD*). For security purposes the document is initially encrypted. Using our proposed scheme, *fingerprinted secret sharing steganography*, the encrypted document can now be secretly hidden in an image. The image can be chosen by the President of *WEST FIELD*. It could be a family photo or a party photo, i.e. any photographic image that would not rouse any suspicion. Initially the image is logically divided into *n* sub-images. The confidential document is then embedded in those sub-images. Each sub-image can now be distributed to those employees in *WEST FIELD* who are eligible to gain access to that confidential document. Let the organization policy be that the document can be recovered by (a) any eight Managers, or (b) a CEO and six managers, or (c) two CEOs and a Vice President, or (d) two Vice Presidents, or (e) the President of West Field. Then one could use a (8, n) *fingerprinted secret sharing steganography* scheme to recover the confidential document. The sub-images can be distributed to the employees in the following manner: Each manager gets one sub-image, each CEO gets two, each Vice President gets four and the President gets eight. This confidential document can be recovered whenever eight sub-images are available.

III. IMAGE STEGANOGRAPHY

Simmons first introduced the concept of steganography in the early 1980s when he discussed the prisoners' problem [18]. He discussed the situation in which two prisoners who are locked in different cells have to communicate covertly in order not to raise suspicion. Simmons used the idea of subliminal channels instead of steganography. This was one of the first works in the field of steganography.

The literature shows the existence of a variety of techniques in use in which data can be embedded in images [10, 11, 12, 13, 14, 15, 16, 17, 20]. At the same time, this literature also shows the existence of several techniques being used in which hidden data can be detected [1, 2, 3, 4, 5, 6, 7, 9, 21]. Image steganographic techniques can be classified on the basis of the domains in which data is embedded. Basically there are two domains: the spatial domain and the transform domain. Steganographic techniques try to embed data in these domains.

In the spatial domain image steganography the simplest technique is to embed data in the least significant bit (LSB) of each pixel in the cover image. The LSB Replacement technique alters the insignificant information in the cover image. It places the embedding data at the least significant bit (LSB) of each pixel in the cover image. There are two types of LSB insertion methods: fixed-sized and variable-sized. The former embeds the same number of message bits in each pixel of the cover-image, whereas the latter embeds a random number of bits per pixel.

Recently, some steganographic techniques have been reported which directly modify the pixels to embed data.

Some of them are reported here [19, 22]. Wu et al. (2003) proposed the pixel value differencing (PVD) method of steganography, which can hide large amount of data by modifying the different values between pairs of adjacent pixels. Using this technique, more data can be inserted into areas where differences in the adjacent pixel values are large, as pixels in these areas can tolerate more changes and this leads to good imperceptibility and a high embedding rate. Potdar et al. (2004) showed how data can be directly embedded in the spatial domain of images by directly modifying the absolute values of pixels.

In the transform domain, data can be hidden by modifying the Discrete Cosine Transform (DCT) coefficient values. These techniques are normally applicable to JPEG images because JPEG images are stored as DCT coefficient values. There are several algorithms that modify these DCT coefficient values to hide data. The algorithm made by Derek Upham [8] was one of the first algorithms that embedded data in the frequency domain of JPEG images by modifying the DCT coefficient values. It offered an embedding capacity of 12.8% of the steganogram's size. But it was detected by chi-square test proposed by Westfeld [21]. The chi-square test proposed by Westfeld could only detect sequentially embedded messages. Later Provos (2001) proposed the Outguess algorithm to counter the statistical chi square test based on frequency counts and also offered and extended chi-square test that could detect randomly embedded messages. They also showed that their algorithm is not detected using the extended chi-square test. They observed that for JPEG images the fraction of redundant bits that can be used to hold the hidden message does not increase linearly for images with more DCT coefficients. Another algorithm (F5) proposed by Westfeld [20] addresses the weaknesses inherent in the Outguess algorithm. This algorithm modified the absolute values of the DCT coefficients instead of modifying its LSB values. It uses matrix encoding and permutative straddling to reduce the number of steganographic changes. As a result, this is resistant to the chi-square test as well as offering more data embedding capacity compared to Outguess. A more recent work by Sallee presents an information-theoretic method for steganography, which is termed as Model-Based Steganography. It offers high data embedding capacity as well as being resistant against statistical attacks [17].

All the techniques discussed above either try to provide high data embedding capacity or resistance against statistical detection, but we have found very few researchers who have offered breakthrough thinking in tackling the issues of image cropping. Here we propose to use the concept of secret sharing to tackle this issue. We make logical pieces of an image and hide chunks of data within each piece. Since the only way to tackle cropping issues in image is to replicate data, we must pre-process the data to reduce the amount of data that we need to embed. A detailed discussion is provided in the next section.

IV. NEW TECHIQUE- FINGERPRINTED SECRET SHARING STEGANOGRAPHY

In this section, we propose our conceptual framework. The technique we propose is termed as fingerprinted secret sharing steganography. The basic logic behind the proposed techniques is to logically break down an image into multiple sub-images and use those sub-images to embed segmented confidential data. The confidential data is initially pre-processed, i.e. encrypted and compressed. The confidential data is now split into multiple data segments of equal size, and each of these segments undergoes mathematical processing and is finally embedded in the sub-images. Here we use secret sharing scheme to process the data segments. The processed data segments are now encrypted using the intended recipient's public key and finally embedded in the sub-images. The embedding can be done using any steganographic algorithm. A fingerprint function is now applied to the stego sub-images to check the integrity of the stego samples. The fingerprints are delivered to the combiner, a person who plays the role of regenerating the confidential data. Before describing our conceptual framework in further detail, we start by introducing some elementary concepts. This is followed by the technical overview, algorithm description and examples.

A. Lagrange Interpolating Polynomial

Our scheme is based on polynomial interpolation: given k points in the 2-dimensional plane $(x_1, y_1), ... (x_k, y_k)$ with distinct x_i, there is one and only one polynomial $q(x)$ of degree $(k-1)$ such that $q(x_1) = y_i$ for all i. The polynomials can be replaced by any collection of functions that are easy to evaluate and to interpolate. Here we use the Lagrange Interpolating Polynomial[1]. The Lagrange interpolating polynomial is the polynomial of degree $(n-1)$ that passes through the n points $y_1 = f(x_1), y_2 = f(x_2), ... y_n = f(x_n)$. It is given by

$$P(x) = \sum_{j=1}^{n} P_j(x) \quad \text{Where} \quad P_j(x) = y_j \prod_{\substack{k=1 \\ k \neq j}}^{n} \frac{x - x_k}{x_j - x_k} \quad (1)$$

Written explicitly,

$$P(x) = \frac{(x - x_2)(x - x_3)...(x - x_n)}{(x_1 - x_2)(x_1 - x_3)...(x_1 - x_n)} y_1 +$$

$$\frac{(x - x_1)(x - x_3)...(x - x_n)}{(x_2 - x_1)(x_2 - x_3)...(x_2 - x_n)} y_2 + ...$$

$$\frac{(x - x_1)(x - x_2)...(x - x_{n-1})}{(x_n - x_1)(x_n - x_2)...(x_n - x_{n-1})} y_n \quad (2)$$

For n = 3 points

[1] Lagrange Polynomial
   http://icl.pku.edu.cn/yujs/MathWorld/math/l/l029.htm

$$P(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)} y_3$$

$$P'(x) = \frac{2x - x_2 - x_3}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{2x - x_1 - x_3}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{2x - x_1 - x_2}{(x_3 - x_1)(x_3 - x_2)} y_3$$

$$(3)$$

Note that the function $P(x)$ passes through the points (x_i, y_i), as can be seen for the case $n = 3$,

$$P(x) = \frac{(x_1 - x_2)(x_1 - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x_1 - x_1)(x_1 - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x_1 - x_1)(x_1 - x_2)}{(x_3 - x_1)(x_3 - x_2)} y_3 = y_1$$

$$P(x) = \frac{(x_2 - x_2)(x_2 - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x_2 - x_1)(x_2 - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x_2 - x_1)(x_2 - x_2)}{(x_3 - x_1)(x_3 - x_2)} y_3 = y_2$$

$$P(x) = \frac{(x_3 - x_2)(x_3 - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x_3 - x_1)(x_3 - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x_3 - x_1)(x_3 - x_3)}{(x_3 - x_1)(x_3 - x_2)} y_3 = y_3$$

$$(4)$$

Generalizing to arbitrary n,

$$P(x_j) = \sum_{k=1}^{n} P_k(x_j) = \sum_{k=1}^{n} \delta_{jk} y_k = y_j \quad (5)$$

B. Shamir's Secret Sharing Scheme

According to Shamir's Secret Sharing Scheme, a piece of data D is divided into n pieces in such a way that D is easily reconstructable from any k pieces, but even complete knowledge of $k - 1$ pieces reveals absolutely no information about D. The main goal is to divide the data D into n pieces $D_1, ..., D_n$ in such a way that: knowledge of any k or more D_i pieces makes D easily computable but knowledge of any $k - 1$ or fewer D_i pieces leaves D completely undetermined. Such a scheme is termed as (k, n) threshold scheme.

C. Fingerprinted Secret Sharing Steganography

In this section we provide a detailed discussion of our proposed technique. We use the secret sharing scheme proposed by Shamir in the context of image steganography. We apply the concept of fingerprinting to achieve data integrity. The process of information hiding is basically divided into two steps: embedding and extraction. We first describe the embedding algorithm and then discuss the extraction algorithm.

The *embedding algorithm* begins by pre-processing the data D using any data processing algorithm. We define a threshold value k, which would be the minimum number of data segments required to regenerate the data completely.

719

We compute a (k-1)-degree polynomial $F(x)$ to embed the processed data D. The polynomial we choose in our algorithm is $F(x) = D + d_1 x + d_2 x^2 + \cdots + d_{k-1} x^{k-1}$, where $d_1, d_2, \ldots d_{k-1}$ are coefficients. This polynomial will be used to share a secret amongst multiple users. A polynomial $F(x)$ of degree $(k-1)$ is determined by its values at k different points. Therefore, we choose n different values of its argument $x_1, \ldots, x_n$, and then compute $F(x_1), \ldots, F(x_n)$. As a result, we could re-construct F(x) by using k points of $(x_1, F(x_1)), \ldots, (x_n, F(x_n))$. To share the secret we need to calculate n-points $(x_1, F(x_1)), \ldots, (x_n, F(x_n))$. We encrypt the following set $F(x_1), \ldots, F(x_n)$ using the intended recipient's public key $P_1, \ldots, P_n$ to obtain ciphertext $G_1, \ldots, G_n$. We divide an image I into n sub-images $H_1, \ldots, H_n$ and embed $G_1, \ldots, G_n$ into $H_1, \ldots, H_n$ respectively, to generate the stego sub-images $M_1, \ldots, M_n$. To maintain the integrity of the stego sub-images, we apply a fingerprint function $h(\cdot)$ on each of the stego sub-images $M_1, \ldots, M_n$ to obtain $f_1, \ldots, f_n$ and deliver it to the combiner. A Combiner is a person who will play the role of regenerating the confidential data. We give $(x_1, M_1), \ldots, (x_n, M_n)$ respectively to each recipient $P_1, \ldots, P_n$. Since we can regenerate the complete set of data D using k out of the n sub-images, we have a very high probability of data recovery in case the image is tampered with by cropping. This is the complete embedding procedure.

We will now discuss the *extraction algorithm*. The extraction algorithm begins by the combiner inviting all the participants $P_1, \ldots, P_n$ who want to contribute to recovering the shared secret. If $p < k$ the extraction procedure cannot proceed because we need at least k participants to submit their sub-images to successfully recover the shared secret. If we have $p \geq k$ the extraction process goes to the next step. In this step the combiner checks the fingerprints of each sub-image to verify that each sub-image is consistent or whether it has been tampered with. If the fingerprints don't match the extraction process stops, whereas if the fingerprints match it goes to the next step. In this step each participant retrieves its G_{i_j} $(i_j = i_1, \ldots, i_p)$ using the stego key and the steganographic algorithm used. Each participant then recovers $F(x_{i_j})$ using their private key and gives it to the combiner. We recover at least k values of $F(x_{i_j})(1 \leq j \leq p)$ to generate D using Lagrange Interpolating Polynomial. We will use the Equation 1 to calculate $F(x)$ and then set the value of x to zero i.e. calculate $F(0)$ to regenerate D. This is where the extraction steps finish.

Remark 1: We may think of a special scenario where t $(t > n-k)$ sub images were lost, then how could we recover the data? To tackle this situation we add one more level of security in our algorithm. We concatenate the hash of the ciphertext $h(G_i)$ with the ciphertext G_i and then embed $h(G_i) \parallel G_i$ in the sub-image H_i $(1 \leq i \leq n)$. This will have an advantage when a sub-image has been identified as being tampered but we want to check whether the data within that sub-image is still valid.

Remark 2: To achieve high robustness against cropping we suggest setting the value of n as high as possible and the setting the value of k as low as possible. If the following constraint is followed we can recover the data with a very high probability.

Remark 3: To achieve even higher robustness against cropping we suggest embedding G_i $(i < n)$ multiple times. Since the size of G_i is relatively very small compared to H_i $(i < n)$, we can embed G_i without making any remarkable changes to the cover image.

Remark 4: If this scheme is to be applied to pure steganography, the assumption that $x_1, \ldots, x_n$ would be given to each recipient is no longer valid. Hence we suggest embedding x_i $(i < n)$ in H_i $(i < n)$ along with G_i

Remark 5: Data can be represented in the form of 2^a or $2^a + b$. In the example discussed in the Section VI we use the data in the form of 2^a. Representing data in this format gives us phenomenal embedding capacity, and using this method we can hide a large amount of data without making large changes in the cover medium. As a result we get the privilege to duplicate the data in the cover medium. By using this technique we change the focus to data regeneration. Here we rely on the systems processing capability to achieve high embedding capacity. So there is a tradeoff between processing time and embedding capacity.

V. ALGORITHM DESCRIPTION

The steps in the algorithm can be described as under:

A. Embedding Algorithm

1. Preprocess the data D.
2. Choose a suitable number n and set the threshold value to k.
3. Compute a $(k-1)$ degree polynomial $F(x) = D + d_1 x + d_2 x^2 + \cdots + d_{k-1} x^{k-1}$
4. Choose $x_1, \ldots, x_n$ and calculate $F(x_1), \ldots, F(x_n)$
5. Encrypt $F(x_1), \ldots, F(x_n)$ using the public key of the intended recipients i.e. $P_1, \ldots, P_n$ to obtain $G_1, \ldots, G_n$
6. Divide the image into n logical sub-images $H_1, \ldots, H_n$.

720

7. Hide $G_1, ..., G_n$ respectively to $H_1, ..., H_n$ using a steganographic algorithm to obtain $M_1, ..., M_n$.

8. Calculate the fingerprints for the new sub-images $M_1, ..., M_n$ by using a fingerprint function $h(\cdot)$ to obtain $f_1, ..., f_n$ (n fingerprints), and deliver them to the combiner.

9. Deliver $(x_1, M_1), ..., (x_n, M_n)$ to $P_1, ..., P_n$.

B. Extraction Algorithm

1. The combiner finds the number of participants p who would contribute to recover the shared secret, e.g. $P_1, ..., P_n$. If $p < k$, then stops. Otherwise, goes to the next step.

2. The combiner checks the fingerprints on each collected sub-image $M_{i_1}, ..., M_{i_p}$ (from $P_1, ..., P_n$). If all the fingerprints are consistent to these p sub-images respectively, then it goes to the next step. Otherwise, stop the extraction.

3. Each participant retrieves its $G_{i_j} (i_j = i_1, ..., i_p)$ using the stego key and the stego algorithm used.

4. Each recipient then retrieves its $F(x_{i_j})$ value using its private key.

5. Retrieve at least k values of $F(x_{i_j})(1 \le j \le p)$ to regenerate D.

6. Use the Equation 1 to calculate $F(x)$ using Lagrange Interpolation Polynomial.

7. Calculate $F(0)$ to regenerate D.

C. Parameters Specification

In this subsection we will specify the parameters used in the above *Embedding Algorithm* and *Extraction Algorithm*. We just provide some technical explanations related to those parameters. The detailed specifications may be determined when this scheme is applied to some industrial scenarios.

1. The document D will be processed into a smaller file if it is very large. Also, D will be transferred into an integer when we construct the polynomial $F(x) = D + d_1 x + d_2 x^2 + \cdots + d_{k-1} x^{k-1}$.

2. For the selection of k, we follow this principal. The larger the k, the better will be the outcome. However, k is always less than or equal to n. k can be a variable, which can be decided according to individual scenario.

3. The selection of $d_1, d_2, ... d_{k-1}$, will be either random or can be chosen by the user. They are all integers.

4. The image will be logically segmented depending upon the number of data segments and the rate of redundancy required.

5. The values $x_i (1 \le i \le n)$, will be a function of $H_i (1 \le i \le n)$ respectively. For example, $x_i = \left(9 y_{\min} + y_{\max}^2\right) \bmod q \ (1 \le i \le n)$, where $y_{\min}$ and $y_{\max}$ are the minimum value and the maximum value of the pixels of the sub-image.

6. For the creation of the fingerprints on those sub-images, we will use a secure cryptographic hash function $h(\cdot)$, which is collision-free. Therefore, those fingerprints are calculated by $f_1 = h(M_1), \cdots, f_n = h(M_n)$.

7. For the step 2 of the above Extraction Algorithm, the combiner should check the fingerprints on the collected sub-images $M_{i_1}, \cdots, M_{i_p}$. If the fingerprints are not verified, the re-constructed polynomial $F(x) = \underline{D} + \underline{d_1} x + \underline{d_2} x^2 + \cdots + \underline{d_{k-1}} x^{k-1}$ may be not identical to the original one $F(x) = D + d_1 x + d_2 x^2 + \cdots + d_{k-1} x^{k-1}$; since some participants would have modified their sub-images $M_{i_1}, \cdots, M_{i_p}$.

8. The combiner in the Extraction Algorithm could play the role of the President of West Field described in the scenario of Section 2.

VI. PRELIMINARY RESULTS OR EXAMPLE DESCRIPTION

In this section we discuss an example scenario to describe the working of our algorithm, and we shall present some theoretical results. We show how we can hide data D in an image of size 512 x 512. An example of such an image is shown in Fig. 2.

A. Data Preprocessing Step

Before hiding the data we preprocess the data to a format that could be easily processed using the $(k-1)$ degree polynomial. The data is processed and represented in the following format either $D = 2^a$ or $D = 2^a + b$, $(a, b \in \mathfrak{R}^+)$. Once the data is represented in this format we use the coefficient a and b as the input to the $(k-1)$ degree polynomial $F(x)$. In this example scenario we consider the data to be represented in 2^a format e.g. 2^{70} hence the value for a becomes 70. Since we are representing data in this format b is not used in the polynomial and hence its value is not shown in Table 1.

Fig.2 Image of Lena

This data preprocessing step gave us a phenomenal embedding capacity although at a cost of extra processing.

This phenomenal embedding capacity helped us to embed data several times and recover the data in case of cropping. We begin by taking the preprocessed data D and deciding all the necessary coefficients for the $(k-1)$ degree polynomial. The sample details of these coefficient values are shown in Table 1.

Coefficient	Value	Coefficient	Value
x_1	4	d_1	1
x_2	5	d_2	3
x_3	7	d_3	11
x_4	9	k	4
x_5	11	n	8
x_6	13	a	70
x_7	17	b	Not Used
x_8	19		

Table 1: List of Coefficients used in calculating Sample Results

B. Secret Sharing Step

While calculating the exemplar results we set the threshold value i.e. $(n = 8)$ and $(k = 4)$, which means we split the secret into 8 pieces using the $(k-1)$ degree polynomial, and we require at least 4 secret pieces to recover or regenerate the data D completely.

C. Polynomial Calculation Step

In the example scenario we used a polynomial equation of degree 3 i.e. $F(x) = D + d_1 x + d_2 x^2 + d_3 x^3$. We choose the values of $d_1,....,d_3$ randomly so the equation becomes $F(x) = D + x + 3x^2 + 11x^3$. These random values are shown in Table 1. Since we are dividing the secret into 8 pieces we need to calculate $F(x)$ eight times to distribute it to 8 recipients. Hence we need to decide the values of $x_1,...,x_8$. These values can be chosen by the user, but it should satisfy one constraint, i.e. the GCD of any two of those numbers should be 1. The selected values for $x_1,...,x_n$ satisfy this constraint and are shown in Table 1. The output of $F(x_1),...,F(x_8)$ is shown in Table 2. This output $F(x_1),...,F(x_8)$ along with $x_1,...,x_8$ is now embedded in the image using any publicly available steganographic algorithm.

D. Image sub-division Step

Before the embedding process can begin the image is first logically divided into n sub-images, in this case we divide the main image into 8 sub-images. Each of these sub-images would be used to hide one value from $F(x_1),...,F(x_8)$ and/or $x_1,...,x_8$ depending upon how the technique is used. Once the outcome of the polynomial is embedded in the im-

age we do a simple cropping test on the image to identify how much cropping the image can bear to recover the data completely. The sample result is shown in Figure 3.

$(k-1)$ **Degree Polynomial Equation**
$F(x) = D + d_1 x + d_2 x^2 + d_3 x^3$
$F(4) = 70 + (4) + 3(4)^2 + 11(4)^3 = 826$
$F(5) = 70 + (5) + 3(5)^2 + 11(5)^3 = 1525$
$F(7) = 70 + (7) + 3(7)^2 + 11(7)^3 = 3997$
$F(9) = 70 + (9) + 3(9)^2 + 11(9)^3 = 8341$
$F(11) = 70 + (11) + 3(11)^2 + 11(11)^3 = 15085$
$F(13) = 70 + (13) + 3(13)^2 + 11(13)^3 = 24757$
$F(17) = 70 + (17) + 3(17)^2 + 11(17)^3 = 54997$
$F(19) = 70 + (19) + 3(19)^2 + 11(19)^3 = 76621$

Table 2: List of Polynomial Equation Outcomes

E. Result Discussion Step

Here we have shown that if image is logically divided into 8 sub-images (i.e. $(n = 8)$) and the threshold value for data recovery is 4 (i.e. $(k = 4)$), then we can recover the data even if the image is cropped by 50%, provided the image is cropped in a uniform manner (i.e. if we loose 4 sub-images we can still recover the complete data without any loss.). As outlined in Remark 2, if we keep the value of n very high and set the value of k comparatively low we can achieve even better results; i.e. we can decide how much cropping we can expect and accordingly set the value for k and n. In the above example the difference between n and k is 4 units, so we can recover data if the image is cropped by 50%. If we set $(n = 16)$ and $(k = 4)$ we can theoretically recover complete data even if 75% of image is cropped.
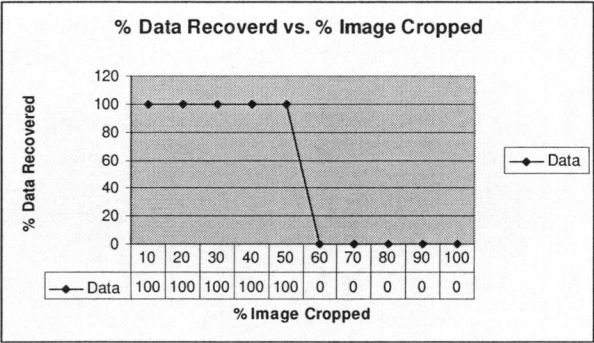

% Data Recoverd vs. % Image Cropped									
	10	20	30	40	50	60	70	80	90	100
Data	100	100	100	100	100	0	0	0	0	0

Fig.3 Results

F. Data Recovery Step

When recovering the data we first access each sub-image and try to recover at least four unmodified sub-images. Once the sub-images are accessed the steganographic algorithm recovers the hidden data, i.e. the values of $F(x_1),...,F(x_8)$ and $x_1,...,x_8$. Once these values are recovered we use only four of these values to generate $F(0)$ from Eq. 2. By using this equation we find $F(0)$ and the output of this equation represents D, because when we set the value of x to zero in $F(x) = D + d_1 x + d_2 x^2 + d_3 x^3$ the result is D. That is how we recover D. Thus by using Lagrange Interpolating Polynomial we can easily recover D.

Remark 6: If this technique is used purely for secret sharing, where the image is physically broken down to n pieces, we don't need to embed $x_1,...,x_8$ because it would be assumed that this set would be available with either the combiner or the user. But in case this technique is purely applied for steganographic use then we suggest embedding $x_1,...,x_8$ along with the actual secret data.

VII. CONCLUSION

In this paper we have applied the concept of Fingerprinted Secret Sharing Scheme to the Steganography Domain. We have shown how we can use this technique to tackle the issue of data loss by image cropping. We used the Lagrange Interpolating Polynomial to process the secret data before embedding it in an image. The security of our scheme is mainly based on the secure principle of steganography and secret sharing scheme. The technique of Fingerprinted Secret Sharing Steganography is shown to be effective against the image cropping problem. We have shown how it can be efficiently used to recover data from a cropped image.

VIII. REFERENCES

[1] Avcibax I., N. Memon and B. Sankur, 2003. "Steganalysis using image quality metrics" *IEEE Transactions on Image Processing*, vol. 12, no. 2, Feb 2003, pp. 221-230

[2] Ira S. Moskowitz, LiWu Chang, Richard E. Newman 2002. Capacity is the wrong paradigm. *In Proceedings of the 2002 workshop on New security paradigms*. pp.114 – 126, Virginia Beach, Virginia

[3] Fridrich, J., "Feature Based Steganalysis for JPEG Images and its Implications for Future Design of Steganographic Schemes" 2004. In *Proc. 6th Information Hiding Workshop*, Toronto, Canada, May 23-25, 2004.

[4] Fridrich, J., Goljan, M., Hogea, D., Soukal, D., "Quantitative Steganalysis of Digital Images: Estimating the Secret Message Length" 2003. In *ACM Multimedia Systems Journal, Special issue on Multimedia Security*, Vol. 9(3), 2003, pp. 288-302

[5] Fridrich, J., Goljan, M., Hogea, D., 2002. "Attacking the Outguess", *Proc. of the ACM Workshop on Multimedia and Security*, Juan-les-Pins, France, December 6, 2002.

[6] Fridrich, J., Goljan, M., Hogea, D., 2002. "Steganalysis of JPEG Images: Breaking the F5 Algorithm", 5th Information Hiding Workshop, Noordwijkerhout, The Netherlands, 7–9 October 2002, pp. 310-323.

[7] Fridrich, J., Goljan, M., Du R., 2000. "Steganalysis based on JPEG Compatiblity Steganalysis", *Special session on Theoretical and Practical Issues in Digital Watermarking and Data Hiding, SPIE Multimedia Systems and Applications IV*, Denver, CO, August 20-24, 2001, pp. 275-280.

[8] Derek Upham, 1999. "Jsteg Steganographic Algorithm" Available on the internet ftp://ftp.funet.fi/pub/crypt/steganography/

[9] Johnson, N. F., Jajodia, S., 1998. Steganalysis of images created using current steganographic software. *In*: D. Aucsmith, ed. *2nd International Workshop on Information Hiding April 14-17, 1998, Portland, Oregon, USA*. Springer, 273-289.

[10] Khan, M., Potdar V, Chang E., 'A prototype implementation of Grey Level Modification Steganography', in *Proceedings of the 30th Annual Conference of the IEEE Industrial Electronics Society*, IECON 04, Korea.

[11] Lee, Y. K. & Chen, L. H., 2000. High Capacity Image Steganographic Model. *In IEE Proceedings Vision, Image and Signal Processing*, vol. 147 no. 3, pp. 288-294.

[12] Newman, R. E., Moskowitz, I. S., Chang, L., Brahmadesam M. M., 2002 " A Steganographic Embedding Undetectable by JPEG Compatibility Steganalysis". In: *Petitcolas (Ed.): Information Hiding, 5th International Workshop, IH 2002, Noordwijkerhout, The Netherlands, October 7-9, 2002 LNCS Springer Verlag 258-277.*

[13] Potdar V, Chang E. 'Covering Encrypted Information using Images', European and Mediterranean Conference on Information Systems (EMICS2004), Tunis, Tunisia, July 25-27, 2004

[14] Potdar V, Chang E. 'Hiding Text Cryptography using Image Cryptography', 4th International Networking Conference, Plymouth, U.K. July 6-9, 2004

[15] Potdar V, Chang E. 'Grey Level Modification Steganography for Secret Communication', *2nd IEEE International Conference on Industrial Informatics (INDIN2004)*, Berlin, Germany, June 24-26, 2004

[16] Provos, N., "Defending Against Statistical Steganalysis" 2001. In *Proceedings of the 10th USENIX Security Symposium*, pages 323-335, August 2001

[17] Sallee, P., "Model-Based Steganography" 2003. In International Workshop on Digital Watermarking, Seoul, 2003,

[18] Simmons, G. J., 1984. " The prisoner's problem and the subliminal channel" In *Advances in Cryptology -- CRYPTO '83"*, D. Chaum, ed., Plenum Press, 1984, 51-67.

[19] Soo-Chang, P, Jing-Ming, G., 2003. Hybrid pixel-based data hiding and block-based watermarking for error-diffused halftone images. *In IEEE Transactions on Circuits and Systems for Video Technology.* 13(8), 867- 884.

[20] Westfeld, A., 2001. "High Capacity Despite Better Steganalysis (F5-A Steganographic Algorithm)", In: *Moskowitz, I.S. (eds.): 4th International Workshop on Information Hiding*, LNCS, Vol. 2137. Springer-Verlag, New York, pp. 289--302, 2001.

[21] Westfeld, A., Pfitzmann A., 2000. "Attacks on Steganographic Systems Breaking the Steganographic Utilities EzStego, Jsteg, Steganos, and S-Tools—and Some Lessons Learned". Lecture Notes in Computer Science, vol.1768, Springer-Verlag, Berlin, 2000, pp.

[22] Wu, D.C. and Tsai, W.H., 2003. A steganographic method for images by pixel-value differencing. *In Pattern Recognition Letters.* 24(9-10), pp. 1613-1626.

Flow Charts for the Embedding and Extraction Algorithm

Flow Chart for the Embedding Algorithm

Begin
↓
Pre-process Data D
↓
Choose a suitable value n and set the threshold value to k
↓
Compute $k-1$ degree polynomial using $F(x) = D + d_1 x + d_2 x^2 + \cdots + d_{k-1} x^{k-1}$
↓
Choose x_1 to x_n and calculate $F(x_1), \cdots, F(x_n)$
↓
Encrypt $F(x_1), \cdots, F(x_n)$ with public key $P_1 \ldots P_n$ of the recipient to obtain $G_1 \ldots G_n$
↓
Divide Image I into n logical sub-images $H_1, \cdots, H_n$
↓
Hide $G_1, \cdots, G_n$ respectively to $H_1, \cdots, H_n$ using a steganographic algorithm to obtain $M_1, \cdots, M_n$
↓
Calculate the fingerprints for the new sub-images $M_1, \cdots, M_n$ by using a fingerprint function $h(.)$ to obtain $f_1, \cdots, f_n$, and deliver them to the combiner.
↓
Deliver $(x_1, M_1), \cdots, (x_n, M_n)$ to $P_1 \ldots P_n$.
↓
End

Flow Chart for the Embedding Algorithm

Flowchart for the Extraction Algorithm

Begin
↓
Check the number of participant's p
↓
Is $p<k$ — YES → End
↓ NO
Check the fingerprints on the collected sub-images
↓
Finger-prints consistent — NO → End
↓ YES
Get G_{i_j} using the stego key
↓
Retrieve $F(x_{i_j})$ using the recipient's private key
↓
Retrieve at least k values of $F(x_{i_j})$
↓
Calculate $F(x)$ using Equation 1 and recover D by calculating $F(0)$
↓
End

Flowchart for the Extraction Algorithm