2020

Digital Image Exchange using a No-key(s) Protocol with Phase-only Encryption,

Jonathan Blackledge
Technological University Dublin, jonathan.blackledge@tudublin.ie

N. Mosola
University of KwaZulu-Natal

Digital Image Exchange using a No-key(s) Protocol with Phase-only Encryption

J. M. Blackledge
School of Electrical and Electronic Engineering,
Technological University Dublin;
Department of Computer Science,
University of Western Cape.
jonathan.blackledge@tudublin.ie

N. Mosola
School of Mathematics, Statistics and Computer Science,
University of KwaZulu-Natal;
Department of Computer Science,
National University of Lesotho.
nn.mosola@nul.ls

***Abstract*—This paper considers an algorithm for transferring a digital image over an open network using a No-key(s) Protocol or Three-Way Pass and phase-only encryption/decryption. After providing a short study on the theoretical background to the method, an algorithm is presented on a step-by-step basis. Cryptanalysis is undertaken for the three intercept and single intercept cases, when it is assumed that the encrypted data is intercepted in its entirety for each pass or for any single pass, respectively. The algorithm focuses on the exchange of a JPEG image although in principle, the approach is independent of the format of the image file that is used. Prototype MATLAB functions are provided for the validation of the approach and for further development by interested readers.**

***Index Terms*—Stochastic Phase Functions, Phase-only Encryption, Digital Image Exchange, No-key(s) Protocol, Cryptanalysis.**

I. Introduction

In [1], a method of exchanging a plaintext over an open network was investigated which was predicated on using phase-only encryption to implement a three-pass protocol. In this paper, we consider an extension to this method for exchanging full colour digital images, specifically, and, by way of a commonly used example, JPEG images. The purpose of this is two-fold: (i) to provide an algorithm of exchanging a (full colour) covertext image prior to applications in the method of 'information hiding with data diffusion using convolutional encoding for super-encryption' considered in [2]; (ii) To substantially increase the computational efficiency associated with an application of the algorithm considered in [1] for gray-level or colour images.

After introducing the theoretical background to the problem, we develop a phase-only encryption/decryption algorithm that is used to implement a three-pass protocol. A prototype MATLAB function is provided in the Appendix upon which the results given are based, using tagged image files to pass the encrypted images. The purpose of this is to retain the image data in floating-point form, the accuracy of which (in terms of the number of bits used to represent a floating point number) is fundamental to the numerical performance and successful operation of the algorithm. This is because data integrity is lost during conversion from a floating point array to a bitmap irrespective of the image format that is used.

We consider a method of decrypting the encrypted data by a third party when access to all three passes is available with full data integrity. It is shown that an attack can be prevented by forcing the power spectrum of the cipher for the second pass to become singular. A further approach to breaking a cipheretext is considered using the potential associated with the application of phase-retrieval algorithms. However, it is concluded that unless the image is sparse and without prior information on the plaintext (which unlike repetitive letters, words and phases that occur in text, is not available in an image), this form of cryptanalysis is redundant. Thus, the algorithm presented in this paper provides an accurate and numerically efficient way of exchanging digital images over an open network that does not require the sharing of a key and is computationally difficult an attack.

II. Convolutional Encryption using Phase-only Functions

For an image function $f(\mathbf{r}), \mathbf{r} \in \mathbb{R}^2$ (the image plaintext) and stochastic function or cipher $n(\mathbf{r})$ say, convolutional encryption (or coding) is based on computing the image ciphertext $c(\mathbf{r})$ given by

$$c(\mathbf{r}) = n(\mathbf{r}) \otimes f(\mathbf{r}) \equiv \int_{-\infty}^{\infty} n(\mathbf{r}-\mathbf{s}) f(\mathbf{s}) d^2\mathbf{s} \qquad (1)$$

where $\otimes$ denotes the (two-dimensional) convolution integral. Given Equation (1), it is clear that in order to decrypt $c(\mathbf{r})$ and recover the plaintext $f(\mathbf{r})$, it is necessary to deconvolve $c(\mathbf{r})$ given the cipher $n(\mathbf{r})$. Application of a phase-only cipher provides an exact and unique solution to this problem. This is compounded in the following theorem.

Theorem 2.1: If $n(\mathbf{r})$ has a phase-only spectrum, then the deconvolution problem associated with Equation (1) is well-posed, i.e. $f(\mathbf{r})$ can be recovered from $c(\mathbf{r})$ exactly and uniquely.

Proof 2.1: Let $\Theta(\mathbf{k})$ be the phase function of a unit amplitude phase-only spectrum such that

$$\exp[i\Theta(\mathbf{k})] = N(\mathbf{k}) \leftrightarrow n(x)$$

978-1-7281-9418-9/20/$31.00 © 2020 IEEE

where $\leftrightarrow$ denotes transformation to Fourier or $\mathbf{k}$-space. Applying the convolution theorem to Equation (1), we can write

$$C(\mathbf{k}) = \exp[i\Theta(\mathbf{k})]F(\mathbf{k})$$

where $C(\mathbf{k}) \leftrightarrow c(\mathbf{r})$ and $F(\mathbf{k}) \leftrightarrow f(\mathbf{r})$ and it is then clear that

$$\exp[-i\Theta(\mathbf{k})]C(\mathbf{k}) = F(\mathbf{k})$$

Hence, using the correlation theorem, the plaintext is given by

$$f(\mathbf{r}) = n^*(\mathbf{r}) \odot c(\mathbf{r}) \equiv \int_{-\infty}^{\infty} n^*(\mathbf{r}+\mathbf{s})c(\mathbf{s})d^2\mathbf{s}$$

where $\odot$ denotes the correlation integral

Remark 2.1: The result compounded in *Theorem 2.1* is of no intrinsic value to the deconvolution problem in general which occurs in the applications of digital image processing, for example. This is because it can rarely, if ever, be assumed that the Point Spread Function is characterised by a phase-only spectrum. Further, natural noise can not, in general, be assumed to be characterised by phase-only functions. Thus, it should be understood that *Theorem 2.1* is strictly only applicable to the convolution model given in Equation (1) when $n(\mathbf{r})$ has a phase-only spectrum and thereby, has no applicability to digital image processing in general. However, as explored in this paper, *Theorem 2.1* does have applications in the area of cryptography. This is because the plaintext $f(\mathbf{r})$ can be recovered exactly and uniquely from the diffused plaintext $n(\mathbf{r}) \otimes f(\mathbf{r})$ provided the cipher $n(\mathbf{r})$ is known precisely.

III. Encryption using Phase-Only Stochastic Functions

Consider the encryption model given by Equation (1) where

$$n(\mathbf{r}) \leftrightarrow \exp[i\Theta(\mathbf{k})],\ \Theta(\mathbf{k}) \in [-\pi, \pi]$$

The function $n(\mathbf{r})$ is taken to be a cipher generated by some key dependent algorithm characterised by a phase-only spectrum with a random phase function $\Theta(\mathbf{k})$. For a plaintext function $f(\mathbf{r})$ that is real, the ciphertext is taken to be given by $\mathrm{Re}[c(\mathbf{r})]$. In Fourier space, Equation (1) is

$$C(\mathbf{k}) = F(\mathbf{k})\exp[i\Theta(\mathbf{k})] \tag{2}$$

Let the stochastic phase function $\Theta(\mathbf{k})$ be conditioned to have a uniform distribution and wrapped between $-\pi$ and π radians. The phase function $\Theta(\mathbf{k}) \in [-\pi, \pi]\ \forall \mathbf{k}$ is constructed by taking the Fourier transform of a random (uniformly distributed) variable $s(\mathbf{r}) \in [0, 1]$, say, i.e.

$$\Theta(\mathbf{k}) = \mathrm{atan2}[S(\mathbf{k})],\ \ S(\mathbf{k}) \leftrightarrow s(\mathbf{r}) \tag{3}$$

where $\mathrm{atan2}$ is taken to yield the 4-quadrant phase values in the range $[-\pi, \pi]$ by computing one unique arc tangent value in which the signs of both arguments are used to determine the quadrant of the result, thereby selecting the desired branch of the arc tangent. Since $s(\mathbf{r})$ is real and has no symmetry, $S(\mathbf{k})$ has a symmetric real component and an asymmetric imaginary component. The 4-quadrant phase function is therefore an asymmetric function, i.e. $\Theta(\mathbf{k}) = -\Theta(-\mathbf{k})$.

IV. The Three-pass Protocol

The principle of the Three-pass Protocol is well known as are the algorithms that have been developed for its implementation. These include the Shamir three-pass protocol [5] and the Massey-Omura method [6]. The principle associated with the protocol is as follows: Alice encrypts her plaintext with a known algorithm and private key K_A, say, and sends the ciphertext to Bob. Upon receipt of the ciphertext, Bob cannot decrypt the ciphertext because he does not know K_A. Instead Bob encrypts the ciphertext using the same algorithm but a new private key K_B known only to Bob and sends the now double encrypted plaintext back to Alice. Upon receipt, and critically, assuming the encryption algorithm is commutative, Alice can decrypt the doubly encrypted ciphertext with K_A and send the result (a single encrypted ciphertext) back to Bob who is then able to decrypt the result using K_B. By using this protocol, Alice and Bob do not need to agree upon K_A and K_B *a priori* and thus, no separate key exchange method is required. Three principal conditions for the application of this protocol are required: (i) The encryption algorithm used must be commutative and strong enough so that the ciphertext cannot be broken using a known algorithm attack based on an intercept of any pass, particularly the single encrypted first and third passes; (ii) the keys used must be of a sufficient length to make an exhaustive attack impracticable on any pass; (iii) if the encrypted information is intercepted for each of the three passes, it is not possible to determine the plaintext from the three intercepts (assumed to be complete intercepts in each case). It is the third of the conditions above represents the greatest vulnerability and any encryption system that exploits this protocol must be based on algorithms that exhibit some 'computational difficulty' in this respect. For example, in the case of the Shamir and Massey-Omura algorithms, the security relies on the difficulty of computing discrete logarithms in a finite field [7]. In the following section we consider an application of the three-pass protocol using a phase-only encryption (commutative) algorithm, the cryptanalysis associated with this algorithm being explored in Section VI.

V. Application of Phase-Only Encryption to the Three-Pass Protocol

Consider the case when Alice wishes to exchange an image with Bob which is given by the real two-dimensional function $f(\mathbf{r}) \leftrightarrow F(\mathbf{k})$. Alice generates the random phase cipher $\Theta_1(\mathbf{k})$ and similarly, Bob generates random phase cipher $\Theta_2(\mathbf{k})$. These phase functions are computed through application of the Fourier transform method compounded in Equation (3). The algorithm(s) for generating $s(\mathbf{r})$ from which these phase functions are obtained, are taken to be cryptographically strong and ideally personal to Alice and Bob through application of an evolutionary computing approach [8] including the keys used to seed them. The following steps are then applied.

Step 1: Alice encrypts $F(\mathbf{k})$ to produce ciphertext $C_1(\mathbf{k})$ using the equation

$$C_1(\mathbf{k}) = F(\mathbf{k}) \exp[i\Theta_1(\mathbf{k})] \tag{4}$$

and sends $\mathrm{Re}[c_1(\mathbf{r})]$ of $c_1(\mathbf{r}) \leftrightarrow C_1(\mathbf{k})$ to Bob.

Step 2: Upon receiving the ciphertext $C_1(\mathbf{k}) \leftrightarrow \mathrm{Re}[c_1(\mathbf{r})]$, Bob encrypts $C_1(\mathbf{k})$ using the equation

$$C_2(\mathbf{k}) = C_1(\mathbf{k}) \exp[i\Theta_2(\mathbf{k})] \tag{5}$$

and sends $\mathrm{Re}[c_2(\mathbf{r})]$ of $c_2(\mathbf{r}) \leftrightarrow C_2(\mathbf{k})$ back to Alice.

Step 3: Alice decrypts Bobs ciphertext $C_2(\mathbf{k}) \leftrightarrow \mathrm{Re}[c_2(\mathbf{r})]$ using the equation

$$\begin{aligned} C_3(\mathbf{k}) &= C_2(\mathbf{k}) \exp[-i\Theta_1(\mathbf{k})] \\ &= F(\mathbf{k}) \exp[i\Theta_1(\mathbf{k})] \exp[-i\Theta_1(\mathbf{k})] \exp[i\Theta_2(\mathbf{k})] \\ &= F(\mathbf{k}) \exp[i\Theta_2(\mathbf{k})] \end{aligned} \tag{6}$$

and sends $\mathrm{Re}[c_3(\mathbf{r})]$ of $c_3(\mathbf{r}) \leftrightarrow C_3(\mathbf{k})$ back to Bob.

Step 4: Bob decrypts the ciphertext $C_3(\mathbf{k}) \leftrightarrow \mathrm{Re}[c_3(\mathbf{r})]$ using the equation

$$F(\mathbf{k}) = C_3(\mathbf{k}) \exp[-i\Theta_2(\mathbf{k})] \tag{7}$$

The plaintext is then given by $\mathrm{Re}[f(\mathbf{r})]$ where $f(\mathbf{r}) \leftrightarrow F(\mathbf{k})$. An example for implementing this method for applications in the exchange of JPEG images is given in Section VII.

VI. Cryptanalysis

We consider two approaches to an attack when all three ciphers are intercepted (a three-pass intercept) and when any one cipher is intercepted. In the latter case, we evaluate the use of phase retrieval algorithms in the absence of *a priori* information.

A. Attack based on a Three-pass Interception

Let us assume that an attack is launched to estimate $f(\mathbf{r})$ based on knowledge of the three-pass protocol given in Section V and accurate and complete records of the ciphers $C_1(\mathbf{k})$, $C_2(\mathbf{k})$ and $C_3(\mathbf{k})$ obtained by intercepting the transmission associated with Steps 1-3 and taking the Fourier transform of the results.

Given Equations (4) - (7), we can then eliminate the ciphers Θ_1 and Θ_2 to obtain the equation

$$F(\mathbf{k}) = \frac{C_1(\mathbf{k}) C_2^*(\mathbf{k}) C_3(\mathbf{k})}{\mid C_2(\mathbf{k}) \mid^2}, \mid C_2(\mathbf{k}) \mid > 0 \tag{8}$$

However, since $\mid C_2(\mathbf{k}) \mid^2 = \mid F(\mathbf{k}) \mid^2$ we can write Equation (8) as

$$F(\mathbf{k}) \mid F(\mathbf{k}) \mid^2 = C(\mathbf{k}) \text{ where } C(\mathbf{k}) = C_1(\mathbf{k}) C_2^*(\mathbf{k}) C_3(\mathbf{k})$$

and it is clear that to obtain $F(\mathbf{k})$ we are required to solve a cubic equation, a solution that is given by (obtained using [10])

$$F(\mathbf{k}) = \frac{C_r(\mathbf{k})}{\mid C(\mathbf{k}) \mid^{\frac{2}{3}}} \pm i \left(\frac{C_r(\mathbf{k}) - F_r^3(\mathbf{k})}{F_r(\mathbf{k})} \right)^{\frac{1}{2}} \tag{9}$$

where $C_r(\mathbf{k})$ denotes the real component of the spectrum $C(\mathbf{k})$. Note that because the imaginary component in this solution for $F(\mathbf{k})$ can be either positive or negative, the solution is not unique, leading to ambiguities in this 'plaintext solution'. Moreover, both Equations (8) and (9) have the potential to incur singularities which occur when the amplitude spectrum $\mid C_2(\mathbf{k}) \mid$ approaches zero. Thus, if we 'force' the spectrum $C_2(\mathbf{k})$ to have very low numerical values at the extreme limit of the floating point accuracy available to Alice and Bob, then the effect will be to generate an inverse filter $1/ \mid C_2(\mathbf{k})$ that is singular, thereby eliminating the potential of an attack based on a three-pass intercept. To exploit this effect, we introduce an exponential scaling factor α in the second pass (Step 2) and final decrypt (Step 4) and modify Equations (5) and (7) to the forms

$$C_2(\mathbf{k}) = C_1(\mathbf{k}) \exp[i\Theta_2(\mathbf{k}) - \alpha]$$

and

$$F(\mathbf{k}) = C_3(\mathbf{k}) \exp[-i\Theta_2(\mathbf{k}) + \alpha]$$

respectively. The value of α that is applied depends on the floating point accuracy that is available. In the MATLAB application presented in Section VII and provided in Appednix A, we set $\alpha = 500$ (specifically in the function POX), a value that causes the numerical evaluation of Equation (8) to generate singularities, while maintaining a decrypt that is an identical replica of the plaintext (in a least squares sense), both the plaintext and decrypt being taken to be MATLAB generated JPEG images.

B. Attack using Phase-Retrieval Algorithms

Consider the case when anyone of the ciphertexts in Steps 1-3 is intercepted. From this intercept we can then compute the amplitude spectrum of the plainatext $\mid F(\mathbf{k}) \mid$. This yields a decryption problem that is equivalent to the phase retrieval problem, i.e. given that $f(\mathbf{r}) \leftrightarrow F(\mathbf{k}) = \mid F(\mathbf{k}) \mid \exp[i\Theta(\mathbf{k})]$, then if and only if $\mid F(\mathbf{k}) \mid$ is known, we are required to estimate the phase function $\Theta(\mathbf{k})$ upon which $f(\mathbf{r})$ can be obtained by Fourier inversion. In general, the phase retrieval problem is severely ill-posed with no uniformly stable solutions, a result that holds for frames that are continuous. However, phase estimation algorithms have been developed to provide approximate solutions in two-dimensions, but the uniqueness and stability of such algorithms depends on (i) the sparsity of the image, e.g. $f(\mathbf{r})$ is a binary image; (ii) the existence of *a priori* information on the function $f(\mathbf{r})$ [11]. In regard to the latter case and cryptanalysis, this is equivalent to having a Crib. Although passive Cribs may exist in a written plaintext (in terms of expect key letters, words and phrases, for example), image plaintext's do not generally

have an equivalence. Hence, as long as the plaintext image is not a sparse matrix but a full gray level or colour image, the use of phase-only retrieval algorithms to launch an attack on a single pass intercept is computationally difficult.

VII. Prototype MATLAB Function

Appendix A provides prototype software using MATLAB R2020a to implement the algorithm given in Section V, compounded in the primary function NKP, an acronym for No-Key(s) Protocol. This function relies to three external functions: function POX encrypts/decrypts the data; function WTI writes the ciphertext to a file using a tagged image file format thereby retaining the floating point accuracy associated with the encryption/decryption process; function IF attempts to attack the data and recover the plaintext assuming a three-pass interception. The function NKP has been designed to transfer a .jpg image between two users (Alice and Bob). It encrypts/decrypts the colour components of a full 24-bit colour image separately and has two inputs: (i) The *key* which is a string of numbers between 0 and 9; (ii) the *step* which is assigned input values 1(first pass), 2 (second pass), 3 (third pass) and 4 (for the final decrypt). This key is multiplied by the ASCII decimal integers for the R, G and B (colour components) given by 82, 71 and 66, respectively and is undertaken in function POX in order to provide different keys for the encryption of each colour component, an approach that can be elaborated upon by interested readers. This requires that the length of the key input into function NKP must be a maximum of 7 digits due to the limiting upper bound for a MATLAB random number generator with a non-negative integer seed $< 2^{32}$. The key is used by Alice for the first and third passes. A different key is used by Bob for the second pass and the final decrypt.

Function NKP has been designed on the basis of reading and writing specifically named files. The default input file is 'Image.jpg' which can be any colour jpg image. The RGB colour components of this image are extracted, converted into double precision, the colour image reconstructed and written to the default file 'Plaintext.jpg'. The purpose of this is to ensure that the plaintext is fully compatible with the decrypt within the processing environment that is used, the decrypt being output to default file 'Decrypt.jpg' by running function NKP for *step*=4. For *step*=1, 2, 3, the ciphertext is written to (and read from, for *step*=2 and *step*=3) default file 'Ciphertext_*step*.tif'. It is assumed that these .tif files are sent (by email, for example) from Alice to Bob (*step*=1), from Bob back to Alice (*step*=2) and from Alice back to Bob (*step*=3). After receiving the ciphertext file 'Ciphertext_3.tif', Bob undertakes a final decrypt to recover Alice's plaintext image 'Plaintext.jpg' which is output as a .jpg image to 'Decrypt.jpg'. This step includes a least squares test to evaluate the fidelity of the decrypt with respect to the plaintext (which needs to be known and is consequently a scenario that is not possible in practice) and attempts to recover the plaintext assuming that each pass has been successfully intercepted.

VIII. Discussion and Conclusions

The material presented in this paper is predicated on *Theorem 2.1* which casts the deconvolution problem as well-posed problem and can thereby be solved exactly and uniquely. The solution then depends on knowledge of the phase-only cipher, which in turn, depends on knowledge of the key used to compute the cipher. On the basis of *Theorem 2.1*, we have considered a Three-pass Protocol in order to develop a No-key(s) Protocol algorithm, where Alice and Bob do not need to share their keys. This comes at the 'expense' of passing a different cipherext three times over an open network. The algorithm has been designed to exchange a .jpg image but can be extended to include different image formats as required. Cryptanalysis shows that if the ciphertext is intercepted for each pass, decryption is an ill-posed problem, due to the potential singularities associated with the inverse filter in Equation (8). This characteristic has been exploited by introducing an exponential scaling factor that 'forces' the inverse filter to become singular (within the floating point accuracy that is available), thereby increasing the computational difficulty of an attack. An alternative attack based on the use of phase retrieval algorithms is also computationally difficult providing the image is not sparse and that no *a priori* information on the image (image Cribbs) is available.

Appendix A
Prototype MATLAB Functions for the Implementation of a No-key(s) Protocol using Phase-only Encryption to Exchange a JPEG Image

The functions given in this Appendix are provided to give the reader a guide to the basic programming used to implement the computational procedures discussed in this paper. Both the code and commentary have been condensed in order to comply with the format and prescribed page limit for this publication.

```
function []=NKP(key,step)
%PROCESS: To exchange a full colour jpg
%image using a No-key(s) Protocol (NKP).
%INPUTS: key - an integer string (0-9)
%with a max string length of 7 digits.
%step=1: First pass encryption.
%step=2: Second pass encryption.
%step=3: Third pass decryption.
%step=4: Final decrypt of jpg image.
%EXTERNAL FUNCTIONS:
%POX - Phase-only Encryption/Decryption.
%IF -  Attack using Inverse Filter.
%WTI - Tagged image file (tif) output.
if step==1%Apply processing for step 1.
%Read an image I from a jpg file.
I=imread('Image.jpg');
%Extract RGB components of the colour
%image and convert RGB arrays to double.
I_R =I(:,:,1);I_G =I(:,:,2);
I_B =I(:,:,3);I_R=im2double(I_R);
I_G=im2double(I_G);I_B=im2double(I_B);
```

```
%Reconstruct colour image,
I = cat(3,I_R,I_G,I_B);
%Output new MATLAB generated jpg image
imwrite(I,'Plaintext.jpg','jpg');
%and display the image in figure 1.
figure(1), imshow(I);
%Phase-only encrypt the image,
I=POX(I,key,step);%dispaly in figure 2,
figure(2), subplot(2,2,1),
imshow(I./max(max(I)));
%and output ciphertext to Ciphertext_1.tif.
WTI(I,1); end %Apply second pass.
if step==2 %Read first pass ciphertext,
I=imread('Ciphertext_1.tif');
%phase-only encrypt,
I=POX(I,key,2);%display ciphertext
figure(2), subplot(2,2,2),
imshow(I./max(max(I)));
%and output ciphertext to default tif.
WTI(I,2); end%Apply third pass.
if step==3 %Read second pass ciphertext,
I=imread('Ciphertext_2.tif');
%phase-only decrypt,
I=POX(I,key,3);%display ciphertext
figure(2), subplot(2,2,3),
imshow(I./max(max(I)));
%and output ciphertext to default tif,
WTI(I,3); end %Apply step 4, i.e.
%decryption of third pass ciphertext.
if step==4 %Read third pass cipher.
I=imread('Ciphertext_3.tif');
%Phase-only decrypt,
I=POX(I,key,4);%display in figure 2
figure(2), subplot(2,2,4), imshow(I);
%and output result to Decrypt.jpg.
imwrite(I,'Decrypt.jpg','jpg');
%Compute Least Squares Error between
%Plaintext and Dercypt jpg images.
A=imread('Plaintext.jpg');
B=imread('Decrypt.jpg');
E=sum(abs(A-B).^2,'all')
%Apply 3-pass intercept cryptanalysis.
%Read intercepts.
I_1=imread('Ciphertext_1.tif');
I_2=imread('Ciphertext_2.tif');
I_3=imread('Ciphertext_3.tif');
%Extract RGB components associated with
%each intercept & convert to type double.
%First intercept.
IR_1 =I_1(:,:,1);IG_1 =I_1(:,:,2);
IB_1 =I_1(:,:,3);IR_1=im2double(IR_1);
IG_1=im2double(IG_1);IB_1=im2double(IB_1);
%Second intercept.
IR_2 =I_2(:,:,1);IG_2 =I_2(:,:,2);
IB_2 =I_2(:,:,3);IR_2=im2double(IR_2);
IG_2=im2double(IG_2);IB_2=im2double(IB_2);
%Third intercept.
IR_3 =I_3(:,:,1);IG_3 =I_3(:,:,2);
IB_3 =I_3(:,:,3);IR_3=im2double(IR_3);
IG_3=im2double(IG_3);IB_3=im2double(IB_3);
%Recover plaintext using inverse filter.
[J_R,status_R]=IF(IR_1,IR_2,IR_3);
[J_G,status_G]=IF(IG_1,IG_2,IG_3);
[J_B,status_B]=IF(IB_1,IB_2,IB_3);
%Check status of process.
if status_R==0 | status_G==0 | status_B==0
%and print note that attack has failed.
fprintf('Three-pass intercept Failure\n');
%otherwise reconstruct colour image
else
I = cat(3,J_R,J_G,J_B);
%and display in figure 3.
figure(3),  imshow(I); end; end

function [J]=POX(I,key,step)
%PROCESS:Phase-Only Encrypt/Decrypt.
%INPUTS: Colour Image (I), key & step.
%OUTPUT: Colour Image (J).
%INTERNAL PARAMETER: alpha
alpha=500;%Set value of alpha.
%Extract RGB components, evaluate size
%and convert RGB arrays to type double.
I_R =I(:, :, 1); I_G =I(:, :, 2);
I_B =I(:, :, 3); [N,M]=size(I_R);
I_R=im2double(I_R);  I_G=im2double(I_G);
I_B=im2double(I_B);
%Compute uniformly distributed phase
%arrays for each RGB component using
%the input key and function 'rand'
%based on the 'twister' algorithm.
%Each RGB component key is obtained by
%multiplying the input 'key' with the
%ASCII decimal integers for R G and B.
key_R=key*82;key_G=key*71;key_B=key*66;
rng(key_R,'twister');Theta_R=rand(N,M);
rng(key_G,'twister');Theta_G=rand(N,M);
rng(key_B,'twister');Theta_B=rand(N,M);
%Compute the two-dimensional discrete
%Fourier transforms of each array using
%function 'fft2' and the phase angles
%associated with the real and imaginary
%components of spectrum using function
%'angle' giving random phase-only spectra.
if step==1 | step==2%For step=1 & 2,
    sign=1; end %set sign=1 to encrypt.
if step==3 | step==4%For step=3 & 4,
    sign=-1; end %set sign=-1 to decrypt.
N_R=exp(sign*i*angle(fft2(Theta_R)));
N_G=exp(sign*i*angle(fft2(Theta_G)));
N_B=exp(sign*i*angle(fft2(Theta_B)));
%Set scaling constant for step=2 and
%re-scale data.
```

```
if step==2
c=exp(-alpha);N_R=N_R.*c; N_G=N_G.*c;
N_B=N_B.*c;  end
%Set scaling constant for step=4 and
%re-scale data
if step==4
c=exp(alpha); N_R=N_R.*c; N_G=N_G.*c;
N_B=N_B.*c; end
%Encrypt input RGB components using
%phase only ciphers returning reals
I_R=real(ifft2((fft2(I_R).*N_R)));
I_G=real(ifft2((fft2(I_G).*N_G)));
I_B=real(ifft2((fft2(I_B).*N_B)));
%Reconstruct colour image
J = cat(3,I_R,I_G,I_B);
```

```
function WTI(I,step);
%PROCESS:To write image I to a tif
%maintaining floating point values.
%The function is based on 'Writing an
%image with floating point values', from
%Stackoverflow available at %https://
%stackoverflow.com/questions/14003402/
%writing-an-image-with-floating-point
%-values/33353930
%INPUTS: Image array (I) and step
%OUTPUTS: None
%If step=1 (first pass) write I
%to'Ciphertext_1.tif'
if step==1
t = Tiff('Ciphertext_1.tif','w');end
%If step=2 (second pass) write I
%to 'Ciphertext_2.tif'
if step==2
t = Tiff('Ciphertext_2.tif','w');end
%If step=3 (third pass) write I
%to 'Ciphertext_1.tif'
if step==3
t = Tiff('Ciphertext_3.tif','w');end
t.setTag('Photometric', ...
Tiff.Photometric.RGB);
t.setTag('BitsPerSample', 64);
t.setTag('SamplesPerPixel', 3);
tagstruct.RowsPerStrip = 16;
t.setTag('SampleFormat',...
Tiff.SampleFormat.IEEEFP);
t.setTag('ImageLength',size(I,1));
t.setTag('ImageWidth', size(I,2));
t.setTag('PlanarConfiguration', ...
Tiff.PlanarConfiguration.Chunky);
tagstruct.Software = 'MATLAB';
t.setTag(tagstruct);
t.write(I); t.close();
```

```
function [f,status] = IF(I_1,I_2,I_3)
%PROCESS:Recover plaintext using
%inverse filter for a three-pass attack.
%INPUT: Intercepts for first (I_1),
%second (I_2) and third (I_3) passes.
%OUTPUT: Plaintext f; If the process
%fails, f=0 and status=0, else status=1.
%Compute spectra of ciphertexts.
C_1=fft2(I_1);C_2=fft2(I_2);
C_3=fft2(I_3);
%Compute spectrum of inverse filter,
%checking that all elements > 0.
D=abs(C_2).^2; Check = all(D(:) > 0);
if Check==1
F=(C_1.*C_3.*conj(C_2))./D;
%Return real component of ifft2 and
%set status to value > 0.
f=real(ifft2(F));status=1; else
%Print note on singularity of filter,
fprintf('Singular Inverse Filter\n');
%assign output to 0 and return status=0.
f=0; status=0; end
```

ACKNOWLEDGMENT

The author J. M. Blackledge would like to acknowledge the support of the Science Foundation Ireland, Technological University Dublin, Ireland, University of Western Cape, South Africa, University of KwaZulu-Natal, South Africa and the University of Wales (Wrexham Glyndwr), UK.

REFERENCES

[1] J.M Blackledge, P. Tobin, W. Govere, D. Sibanda and C. M. Adolfo, *Phase-only Digital Encryption using a Three-pass Protocol*, ISSC2019, IEEE UK and Ireland Signal Processing Chapter, Maynooth University, 17-18, June 2019.

[2] J. M. Blackledge, P Tobin, J. Myeza and C. M. Adolfo, *Information Hiding with Data Diffusion using Convolutional Encoding for Superencryption*, International Journal for Pure and Applied Mathematics (Mathematica Aeterna), vol. 7, no. 4, pp. 319-356, 2017.

[3] J. M. Blackledge, *Cryptography and Steganography: New Algorithms and Applications*, Lecture Notes, Center for Advanced Studies, Warsaw University of Technology, Warsaw, 2012, ISBN: 978-83-61993-05-6;https://arrow.dit.ie/engscheleart2/40/

[4] P. C. Mogensen and J. Glückstad, *Phase-only Optical Encryption*, Optics Letters, vol. 15, no. 25(8), pp. 566-568, 2000.

[5] A. Menezes, P. Van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, pp. 500-642, 1996; ISBN: 0-8493-8523-7.

[6] J. L. Massey and J. K. Omura *Method and Apparatus for Maintaining the Privacy of Digital Messages Conveyed by Public Transmission*, US Patent US4567600A, 1986. https://patents.google.com/patent/US4567600

[7] K. Sakurai and H. Shizuya, "A Structural Comparison of the Computational Difficulty of Breaking Discrete Log Cryptosystems", *Journal of Cryptology*, vol. 11, pp. 29-43, 1998.

[8] J. M. Blackledge, S. Bezobrazov, P. Tobin and F. Zamora, "Cryptography using Evolutionary Computing", *Proc. IET ISSC2013*, Letterkenny, Co Donegal, Ireland, June 20-21, 2013.

[9] J. M. Blackledge, *Digital Signal Processing: Mathematical and Computational Methods, Software Development and Applications*, Edition 2, Woodhead Publishing: Series in Electronic and Optical Materials, 2006; eBook ISBN: 9780857099457. https://arrow.dit.ie/engschelebk/4/

[10] Symbolab Algebra Calculator, 2020. https://www.symbolab.com/solver/algebra-calculator

[11] M. V. Klibanov, *On the Recovery of a 2D Function from the Modulus of its Fourier Transform*, Journal of Mathematical Analysis and Applications, vol. 323, no. 2, 818–843, 2006.