Encrypting messages with visual key

DUSMANESCU DOREL
Department for Economical Mathematics and Economical Informatics
Petroleum and Gas University of Ploiesti
Ploiesti, B-dul Bucuresti no. 39, 100680
ROMANIA
doreld@upg-ploiesti.ro

Abstract: - This paper presents a method for encrypting messages, which uses for this operation an image or a part of an image as an encryption/decryption key. The need to encrypt messages for protecting their content knew an increase from the apparition of Internet and electronic transactions. The great number of messages that flows in a computer network and over the Internet is requiring very fast encrypt/decrypt processes for assuring real-time secured communications. Complex algorithms used in present for encrypting/decrypting messages use complex keys and complex encrypt/decrypt functions that need an increased computer power for real-time applications (media).

Key-Words: - message, image, key, encrypting, decrypting, visual key.

1 Introduction

The need to protect information is a major objective for many peoples and organizations. Because every form of transmitting messages can be intercepted, peoples developed techniques to hide the real information in the transmitted messages, transforming them into a message that can not be understood even if it is intercepted.

The explosive increase of Internet and of the number of peoples who use the Internet services has created a universal demand for security, especially for secure high-volume and/or high-speed communications.

Military applications, government, businesses of all sizes and private individuals now have a routine need for protected electronic communications, with a major interest for Internet communications.

One of the Internet applications that need to assure a higher protection for transmitted data is E-commerce (protection of personal data, protection of electronic payments etc.).

One method that is used to protect the transmitted messages is based on cryptography.

The process may be manual, mechanical, or electronic, and the core idea of this paper is to describe some of the many ways in which the encryption process takes place. The process basically consists of a sender and a receiver, a message (called the “plain text”), the encrypted message (called the “cipher text”), and an item used in encryption called a “key.”

The encryption process, which transforms the plain text into the enciphered text, may be thought of as a “black box”. It takes inputs (the plain text and key) and produces output (the encrypted text). The messages may be handwritten characters, electromechanical representations as in a Teletype, strings of 1s and 0s as in a computer or computer network, or even analog speech [4]. The black box will be provided with whatever input/output devices it needs to operate; the interior mechanism, called cryptographic algorithm will, generally, operate independently of the external representation of the information.

The *key* is used to select a specific instance of the encryption process embodied in the machine. It is more properly called the “*cryptovariable*.”

The encrypted text content depends on both the plain text and the cryptovariable. Changing either of the inputs will produce a different output. In typical operation, a cryptovariable is inserted prior to encrypting a message and the same key is used for some period of time [3].

This period of time is known as a “cryptoperiod.” For reasons having to do with obtaining some level of security, the key should be changed on a regular basis. The most important fact about the key is that it embodies most of the security level of the encryption system. By this we mean the system is designed so that complete knowledge of all system details, including specific plain and cipher text messages, is not enough to derive the cryptovariable.

It is important that the system be designed in this fashion because the encryption process itself is seldom secret. The details of the data encryption standard (DES), for example, are widely published so that anyone may implement a DES-compliant system. In order to provide the intended secrecy in the enciphered text, there has to be some piece of information that is not available to those who are not

authorized to receive the message; this piece of information is the cryptovariable, or key.
Cryptography is the domain which studies the means to do encryption. Thus cryptographers design encryption systems. Cryptanalysis is the process of figuring out the message without knowledge of the cryptovariable (key), or more generally, figuring out which key was used to encrypt a whole series of messages.
The cryptography means the ways to encode the initial message using divers techniques, based, mainly, on mathematical and/or logical transformations. The reconstruction of the original message at the destination is also based on mathematical and/or logical transformations.
Before the dawn of the computer era, cryptography used some electromechanical instruments and engines and was based on the human ability to encode or decode messages [1]. The computers offer an increased speed for encoding the messages but, especially, for decoding the secret messages.
Additional, network computers and Internet require special techniques for protecting the messages transmitted between computers.
The last decade witnessed the advent of a great number of complex cryptography algorithms that are implemented for assuring the security of communications in computer networks and in Internet.
Another aspect of these problems consists in the selection of peoples or organizations able to decode the encoded messages for various purposes.
Cryptanalysis is the science of "breaking" the cryptographic systems. In the following we will try to explain what "breaking" a cryptosystem means.
Breaking or attacking a cryptosystem means recovering the plain-text message without possession of the particular cryptovariable (or key) used to encrypt that message.
More generally, breaking the system means determining the particular cryptovariable (key) that was used. Although it is the message (or the information in the message) that the analyst really wants, possession of the cryptovariable allows the analyst to recover all of the messages that were encrypted using that cryptovariable.
Since the cryptoperiod may be days or weeks, the analyst who recovers a cryptovariable will be able to recover many more messages than if he attacks a single message at a time [4].
Determining the specific details of the algorithm that was used to encrypt the message is generally not considered part of breaking an encryption system. In most cases, e.g., DES, the algorithm is widely known.
Even the construction details of many of the proprietary systems such as RC4 and RC5 have been published.
Because it is very difficult to maintain the secrecy of an algorithm it is better to design the algorithm so that knowledge of the algorithm's details is still not enough to obtain the cryptovariable used for a specific message without trying all possible cryptovariables.
Practically, on the market, a continuous competition exists between the cryptography codes creators and decoding algorithms and machines makers for creating unbreakable messages and for breaking these messages.

2. Description of the problem

The problem of communicating a message from a transmitter to a receiver in secured conditions consists in the possibilities of an attacker to intercept the message and view or even modify the information content.

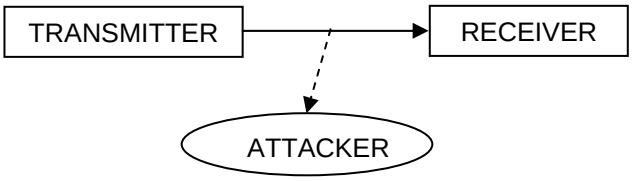


Fig. 1. Intercepting an message

For protecting the content of a message, the transmitter uses cryptographic techniques for modifying the original message making the content not understandable. The main goal is to make sure the attackers will not understand the message even if they intercept them.
The cryptography uses various math functions for encoding the initial message and transforming it into a new message.
For centuries, various forms of encryption have provided confidentiality of information and have become integral components of computer communication technology.
Early encryption techniques were based on shared knowledge between the communication participants. Confidentiality and basic authentication were established by the fact that each participant must know a common secret to encrypt or decrypt the communication, or as with very early encryption technology, the diameter of a stick.
The complexity of communication technology has increased the sophistication of attacks and has raised

the number of vulnerabilities the information exchange is confronting.
The enhancement of communication technology inherently provides tools for attacking other communications.
Therefore, mechanisms are employed to reduce the new vulnerabilities that are introduced by new communication technology. The mechanisms utilized to ensure confidentiality, authentication, and integrity are built on the understanding that encryption alone, or simply applied to the data, will not suffice any longer.
The original message is encrypted and transmitted by an unprotected channel. The receiver decrypts the encrypted message and obtains the original message.
The model of encrypt/decrypt process can be described by the next equations [3]:

$$E(pt) = ct \quad (1)$$
$$D(ct) = pt \quad (2)$$

where *pt* is the plaintext of the message, *ct* – the ciphered text, *E(pt)* – encrypting function and *D(ct)* – decrypting function.
A cryptosystem is composed from the following elements:

- plain text space, *PT*, i.e. the multitude of the possible plain text pt_i;
- keys space, *K*. Every key *k* from *K* induce one encrypting method E_k and one decrypting method D_k;
- cryptotext space, *CT*, i.e. the multitude of possible cryptotexts *ct*. The *CT* elements result from *PT* elements using encrypting method E_k, where *k* is part of *K* domain.

The key represents the link between original message and encrypted message. For a symmetric cryptosystem the key needs to be known by the transmitter and the receiver before establishing communications.
If the key is too simple or all the cryptosystems use simple functions for encrypting messages, there exists the risk that an attacker who intercepts the message decrypts them without knowing the key, by brute force attack. Brute-force attack is the sequential testing of each possible key until the correct one is found. On average, the correct key will be found once half of the total key space has been tried. The only defense against a brute-force attack is to make the key space so huge that such an attack is *computationally infeasible* (i.e., theoretically possible, but not practical given the current cost/performance ratio of computers).
Therefore the safety of a cryptosystem depends on the complexity of the key and on the complexity of the encoding functions. For this reason, today there are used keys with 512, 1024, 2048, 4096 bytes length or even bigger combined with complex encoding functions (polynomial functions, elliptic functions etc.).
The key is combined with the plaintext and computed with a specific algorithm.
There are two primary types of encryption keys: symmetrical and asymmetrical.

2.1. Symmetrical keys

Symmetrical keys are used for both encryption and decryption of the same data.
It is necessary for all the communication participants to have the same key to perform the encryption and decryption. This is also referred to as a shared secret.
In the next example, the transmitter creates a message that is input into an encryption algorithm that uses a unique key to convert the clear message into unintelligible ciphertext.

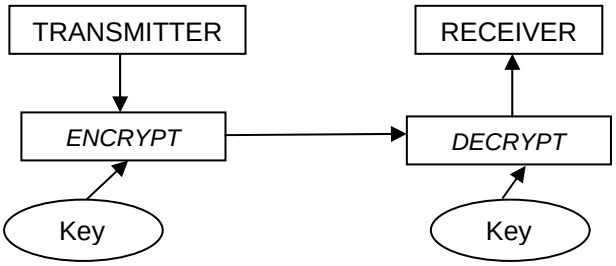


Fig. 2. Cryptography with symmetrical keys

The encrypted result is sent to the receiver, who has obtained the same key through a mechanism called “out-of-band” messaging. The receiver can now decrypt the ciphertext by providing the key and the encrypted data as input for the encryption algorithm. The result is the original plaintext message from transmitter.

2.2. Asymmetrical keys

To further accentuate authentication by means of encryption, the technology of public key cryptography, or asymmetrical keys, can be leveraged to provide message authentication and confidentiality.

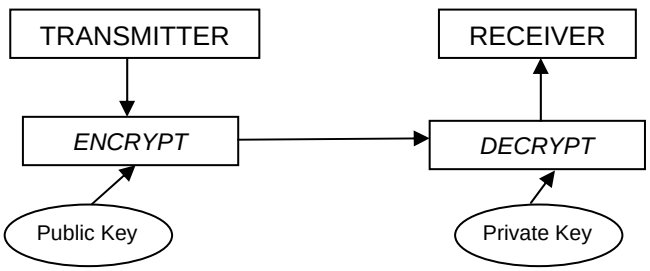


Fig. 3. Cryptography with assymetrical keys

In this case the transmitter and the receiver each maintain a private and public key pair that is mathematically related. The private key is well protected and is typically passphrase protected. The public key of the pair is provided to anyone who wants it and wishes to send an encrypted message to the owner of the key pair.

2.3. Hiding the Data

Another method to protect messages consist in hiding the informations contained in the original message into another message. Recently, there has been an increased interest in steganography (also called stego).

The word *steganography* comes from the Greek, and it means covered or secret writing. As defined today [4], it is the technique of embedding information into something else for the sole purpose of hiding that information from the casual observer.

Many people know a distant cousin of steganography called watermarking — a method of hiding trademark information in images, music, and software.

Watermarking is not considered a true form of steganography. In stego, the information is hidden in the image; watermarking actually adds something to the image (such as the word *Confidential*), and therefore it becomes part of the image.

Some people might consider stego to be related to encryption, but it's not the same thing. We use encryption — the technology to translate something from readable form to something unreadable — to protect sensitive or confidential data. In stego, the information is not necessarily encrypted, only hidden from plain view.

One of the main drawbacks of using encryption is that with an encrypted message — although it cannot be read without decrypting it — it is recognized as an encrypted message.

If someone captures a network data stream or an e-mail that is encrypted, the mere fact that the data is encrypted might raise suspicion. The person monitoring the traffic may investigate why, and use various tools to try to figure out the message's contents.

In other words, encryption provides confidentiality but not secrecy. With steganography, however, the information is hidden; and someone looking at a JPEG image, for instance, would not be able to determine if there was any information within it. So, hidden information could be right in front of our eyes and we would not see it.

In many cases, it might be advantageous to use encryption and stego at the same time. This is because, although we can hide information within another file and it is not visible to the naked eye, someone can still (with a lot of work) determine a method of extracting this information.

Once this happens, the hidden or secret information is visible for him to see. One way to circumvent this situation is to combine the two — by first encrypting the data and then using steganography to hide it. This two-step process adds additional security.

If someone manages to figure out the steganographic system used, he would not be able to read the data he extracted because it is encrypted.

There are several ways to hide data, including data injection and data substitution. In data injection, the secret message is directly embedded in the host medium.

The problem with embedding is that it usuallymakes the host file larger; therefore, the alteration is easier to detect. In substitution, however, the normal data is replaced or substituted with the secret data. This usually results in very little size change for the host file.

However, depending on the type of host file and the amount of hidden data, the substitution method can degrade the quality of the original host file.

In the article "Techniques for Data Hiding", Walter Bender outlines several restrictions to using stego[4]:

• The data that is hidden in the file should not significantly degrade the host file. The hidden data should be as imperceptible as possible.

• The hidden data should be encoded directly into the media and not placed only in the header or in some form of file wrapper. The data should remain consistent across file formats.

• The hidden (embedded) data should be immune to modifications from data manipulations such as filtering or resampling.

• Because the hidden data can degrade or distort the host file, error-correction techniques should be used to minimize this condition.

• The embedded data should still be recoverable even if only portions of the host image are available.

2.4. Steganography in Image Files

As outlined earlier, information can be hidden in various formats, including text, images, and sound files.

To better understand how information can be stored in images, we need to do a quick review of the image file format.

A computer image is an array of points called pixels (which are represented as light intensity and color information).

Digital images are stored in either as 72, 48, 24 or 8 bit pixel files. In a higher color depth image, there is more room to hide information, but these files are

usually very large in size and not the ideal choice for posting them on Web sites or transmitting over the Internet.

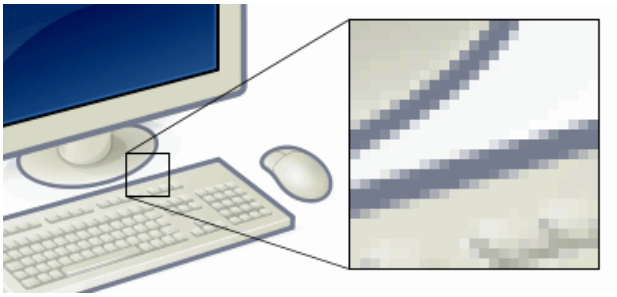


Fig. 4. Digital image composed by pixels

Steganography can be used to hide information in text, video, sound, and graphic files. There are tools available to detect steganographic content in some image files, but the technology is far from perfect. A dictionary attack against steganographic systems is one way to determine if content is, in fact, hidden in an image.

Variations of steganography have been in use for quite some time. As more and more content is placed on Internet Web sites, the more corporations — as well as individuals — are looking for ways to protect their intellectual properties. Watermarking and checksums are methods used to mark documents, and new technologies for the detection of unauthorized use and illegal copying of material are continuously being improved.

In the next section we are presenting a different solution for the encryption problem and for the corresponding algorithm.

3 A different solution for the problem

Consider a communication process like the one in the following figure.

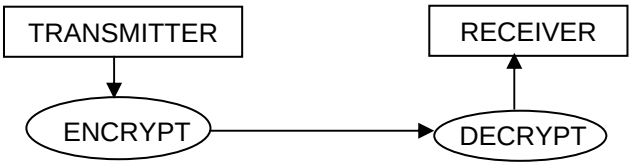


Fig. 5. Encrypted communication process

The communication process consist in the following steps:

- encrypt the original message with the aid of a key K;
- transmit the encrypted message along an unsecured channel for transmitting;
- decrypt the encrypted message with the same key K used by the transmitter.

The force of that process consist in the complexity of keys and the complexity of the encrypting algorithms.

As a key I propose to use an image or a portion of an image defined in electronic format.

As it's shown in the figure number 2 an electronic image is formed by a large number of points named pixels. Every pixel has a single color and the number of pixels from an image determine the resolution of image.

The resolution of an image is a measure of the granularity of the image and is computed as the number of pixels on the horizontal dimension multiplied by the number of pixels on the vertical dimension.

Fig. 6. Image or portion of image used like encrypt key

The color of a pixel is obtained using a color model, usually the RGB model. This mean that the color of every pixel is obtained like a combination between three fundamental colors (Red, Green, and Blue) presented in the pixel color with different intensities. For obtaining a big number of hues for a pixel color we need to use a great number of distinct values for every fundamental color.

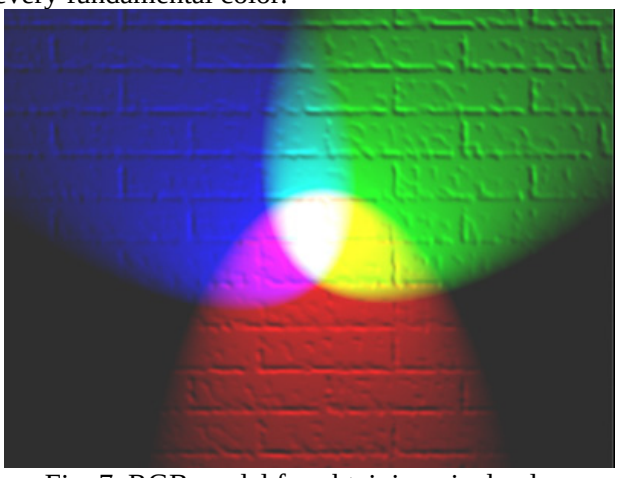


Fig. 7. RGB model for obtaining pixel colors

Today, a true color format for electronic images uses minimum one byte for every fundamental color. Therefore we have available three values between 0-255 for every pixel.
The range of pixels for every row or column from image is unpredictable and has no correlation between two or more pixels. That means that only the entire image can have a meaning and only a part of image and/or a range of pixels may not present a signification by itself.
These characteristics of an image offer the possibilities to use an image or a part of an image like encryption key for encrypting messages.
The power of these algorithms consists in the unpredictable values of ranges of pixels that form the key.

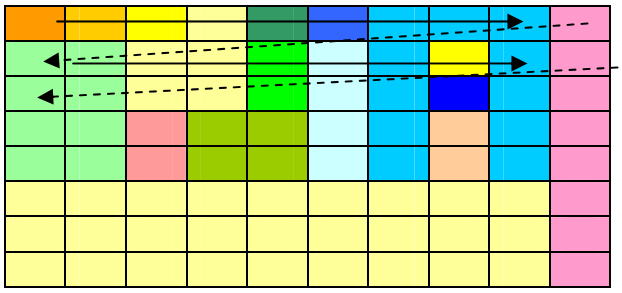


Fig. 8. Image described like an array A

The range of pixels from key doesn't contain a logical or functional link between the pixels values. They are not generated by a logical algorithm, using complex but predictable functions.
Considering the image key like an array, we can use it for encrypting and decrypting the messages by assigning a new value obtained with simple encoding functions for every byte from the original message.
For example, for the array A we have three values for every positions A(i,j).
Considering a digital message like a range of bytes we can transform every byte value, *pt*, from the original message, into a new value, *ct*, using the next algorithm:

IF (A(i,j).Red+*pt*) > 255 THEN
    ct = A(i,j).Red+*pt* – 255
ELSE
    ct = A(i,j).Red+*pt*
IF (A(i,j).Green+*ct*) > 255 THEN
    ct = A(i,j).Green+*ct* – 255
ELSE
    ct = A(i,j).Green+*ct*
IF (A(i,j).Blue+*ct*) > 255 THEN
    ct = A(i,j).Blue+*ct* – 255
ELSE
    ct = A(i,j).Blue+*ct*

The algorithm used for decrypting the encoded message is presented here:

IF (*ct* < A(i,j).Blue) THEN
    ct = *ct* + 255 - A(i,j).Blue
ELSE
    ct = *ct* - A(i,j).Blue
IF (*ct* < A(i,j).Green) THEN
    ct = *ct* + 255 - A(i,j).Green
ELSE
    ct = *ct* - A(i,j).Green
IF (*ct* < A(i,j).Green) THEN
    ct = *ct* + 255 - A(i,j).Green
ELSE
    ct = *ct* - A(i,j).Green

In figure 8 the direction used for covering the entire array A is shown.
An example of byte value *pt* encrypted / decrypted with the previously algorithm is presented here:

pt = 225

A(i,j).Red = 235
A(i,j).Green = 225
A(i,j).Blue = 245

Encrypt:

A(i,j).Red + *pt* = 235+225 = 460 > 255
ct = A(i,j).Red + *pt* – 255 = 460-255 = 205
A(i,j).Green + *ct* = 225+205 = 430 > 255
ct = A(i,j).Green + *ct* – 255 = 430-255 = 175
A(i,j).Blue + *ct* = 245+175 = 420 > 255
ct = A(i,j).Blue + *ct* – 255 = 420-255 = 165

Corresponding values of *pt* is *ct* = 165.

Decrypt:

ct = 165 < A(i,j).Blue = 245
ct = 165 + 255 – 245 = 175
ct = 175 < A(i,j).Green = 225
ct = 175 + 255 – 225 = 205
ct = 205 < A(i,j).Red = 235
pt = 205 + 255 – 235 = 225

As we can see we obtain initial value of *pt* (225).
Another example with other value for *pt* is the following:

pt = 0

A(i,j).Red = 235
A(i,j).Green = 225

A(i,j).Blue = 245

Encryption:

A(i,j).Red + *pt* = 235+0 = 235 < 255
ct = A(i,j).Red + *pt* = 235
A(i,j).Green + *ct* = 225+235 = 460 > 255
ct = A(i,j).Green + *ct* – 255 = 460-255 = 205
A(i,j).Blue + *ct* = 245+205 = 450
ct = A(i,j).Green + *ct* – 255 = 450-255 = 195

Corresponding values of *pt* is *ct* = 195.

Decryption:

ct = 195 < A(i,j).Blue = 245
ct = 195 + 255 – 245 = 205
ct = 205 < A(i,j).Green = 225
ct = 205 + 255 – 225 = 235
ct = 235 = A(i,j).Red = 235
pt = 235 – 235 = 0

We can see that, by decryption, we obtained the initial value of *pt*.
We can verify previous algorithms with a small application written in Delphi. The interface of this application is shown in the next figure.

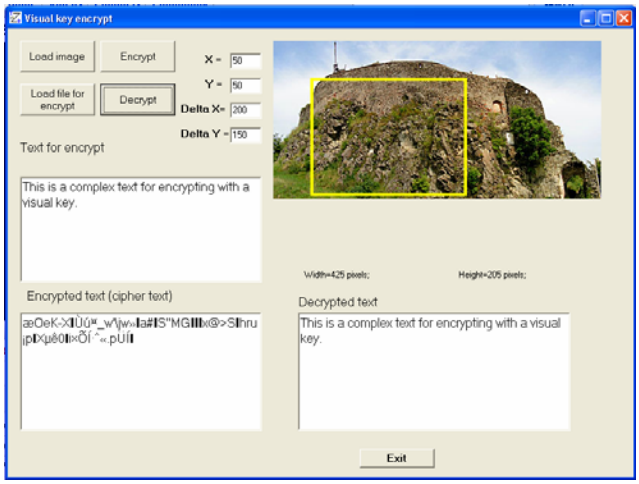
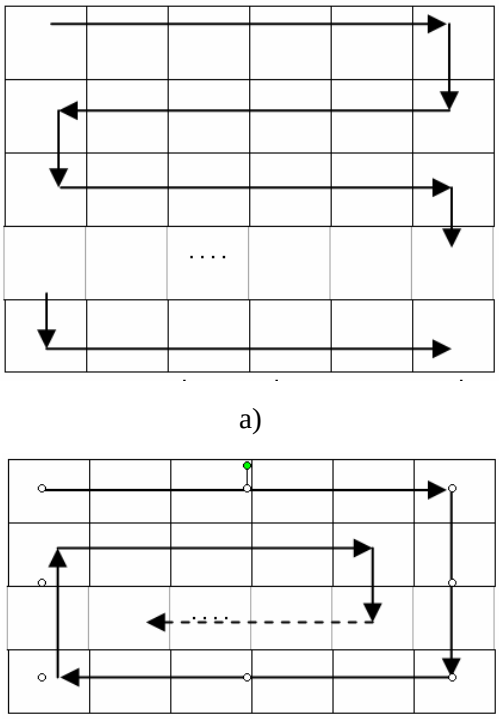


Fig. 9. Interface of the visual key cryptography application

The application make the array key linear for encrypting the entire message. For this operation it is necessary to parse the entire array and storing the values into a vector.
The way specified in Figure 8 is the natural way to walk across an array. We can imagine many ways to do this operation.

In the following figures there are represented some such ways.



a)

b)

Fig. 10. Two simple ways for covering an array

After linearising the array we can encrypt a message with the next algorithm (PASCAL language bassed Delphi code). We suppose that the plain text are stored into a file and the enciphered text will be stored in another file.

```
VAR fin,fout: file of byte; {define the files like byte type}
        pt:byte;
    …
AssignFile(fin,FnamePlainText);
AssignFile(fout,FnameCipherText);
reset(fin);
rewrite(fout);
nvl:=legth(vred);
j:=1;
while not eof(fin) do
  begin
    read(fin,pt);
    if (vred[j]+pt)>255 then
      pt:=vred[j]+pt-255
    else
      pt:=vred[j]+pt;
    if (vgreen[j]+pt)>255 then
      pt:=vgreen[j]+pt-255
    else
```

```
      pt:=vgreen[j]+pt;
    if (vblue[j]+pt)>255 then
      pt:=vblue[j]+pt-255
    else
      pt:=vbluejj]+pt;
    j:=j+1;
    if j>nvl then j:=1;
    write(fout,pt);
  end; {end while}
  closefile(fout);
  closefile(fin);
```

Decryption algorithm is given by the next paragraph:

```
procedure Decrypt;
var sir:string;
     cc:string;
     j:integer;
     fdec,fout:file of byte;
begin
 AssignFile(fdec,FnameCipherText);
 AssignFile(fout,FnameOutClearText);
 reset(fdec);
 rewrite(fout);
 sir:='';
 j:=1;
 while not eof(fdec) do
  begin
    read(fdec,ch);
    if ch<vblue[j] then
      ch:=ch+255-vblue[j]
    else
      ch:=ch-vblue[j];
    if ch<vgreen[j] then
      ch:=ch+255-vgreen[j]
    else
      ch:=ch-vgreen[j];
    if ch<vred[j] then
      ch:=ch+255-vred[j]
    else
      ch:=ch-vred[j];
    j:=j+1;
    if j>nvl then j:=1;
    write(fout,ch);
  end;
  closefile(fdec);
  closefile(fout);
end;
```

As we can observe, the algorithms are very simple and are not based on complex operations, logical or mathematical.
For messages that are bigger than the key vector, after the last value of the key the index *j* resets to the first value of the key.

Obvious, the encrypting and decrypting algorithms can be made more complex, but in this case we will loose the advantage of simplicity.
For increasing the power of these algorithms we can use other ways. For example, changing the mode of array linearization will induce new encryption algorithms.
Additional, using only a part of an image, the logical meaning of the key information is strongly deformed. That will also reduce the chance of an attack.
Another technique which can be used for reducing the risk of decrypting the message by an attack is to apply a graphical filter on the image.
Applying a graphical filter to an image means to increase or decrease the color pixels values in some way or another.
In the next figures the reader can see one such filter applied to an image.

Fig. 11. The original image

Fig. 12. Effect applied on the original image

Usually we can use a unique value for every color value and for every pixel from image key. It is possible to use different color values for apply one filter to an image.
The filter can be applied at the moment of transmittal, with one or more parameters transmitted inside the message.
Receiver can apply the inverse effect and then decrypt the message. Only someone who know the

parameters of the applied filter and have the original image can decrypt the message. Additional, this process can be automated and the distortion parameters can be produced by using a random number generator.
Another power of these method consist in the fact that an image cannot be identical with another if they are not copied having as source the same initial image file.
That means if one attacker want to obtain the key making a photo of the same landscape it is a very low probability to obtain the same image. Practically, because the small differences between the original light and the copy light, the new image differ from the original by many pixel values differences, even if that differences are not visible with the eyes. They exists at the level of the color values information for a lot number of image pixels.

Fig. 13. Region of an image used like encryption key

For using an image like key for encrypting messages it is necessary to specify the portion of image that is used for encrypting the messages.
The pixel coordinates become part of the key together with the horizontal and vertical number of pixel that define that region of image.
With this technique we can grow the level of message security because, even the attackers know the image used like key, if they are not knowing the exact coordinates of the image part selected, they can not decrypt the message.
The encrypted message can include the left corner pixel coordinates (x, y), dx and dy, horizontal and vertical number of pixels that define the region used for the encryption of the message. But, additionally, the message can include other useful information like the parameters of distortion applied to the image and even the image itself.
Using an image with 100 x 100 pixel resolution, the space needed to store it is 32000 bytes (aprox. 30 Kb). For long messages these value do not represent too much. The only problem can be represented by the embeding technique used. But, these can be an advantage because the embedding techniques can offer the way to make the message more complex and more difficult in decryption.

4. Conclusions

The proposed technique for encrypting messages are based on the unpredictable nature of the values of pixels of an image for assuring the security of an encrypted message, making them hard to decrypt by an unauthorized person.
These studies are at the beginning and the entire domain of possibilities offered by these techniques is not yet established. We are researching now their usage for text messages but we disn't studied yet the possibilities to use these techniques for encrypting messages formed by images, movies or sounds
For future development we intend to test these algorithms and try to develop other algorithms that offer more security to communications.
Because the encryption process used in this paper are symmetrical, we consider that the proposed method is fit for private applications. The possibility to use such algorithms in asymmetric encryption processes (applications with public key) can be taken as a subject for future studies.

References:

[1] X1. Claudiu Soroiu, Tehnici de criptare, *GInfo*, No.2, 2002, pp. 30-32
[2] X2. Thomas Cormen, Charles Leiserson, Ronald Rivest, *Introducere în algoritmi*, Editura Libris Agora, Cluj-Napoca, 2000
[3] X3. Patriciu V.V., Pietrosanu-Ene M., Bica I., Vaduva C., Voicu N., *Securitatea comertului electronic*, Editura All, Bucuresti, 2001
[4] X4. Harold F. Tipton, Micki Krause, *Information Security Management Handbook*, Fifth Edition, Auerbach Publications, A CRC Press Company LLC, 2004
[5] X5. Aloka Sinha, Fractional Fourier Transform Based Key Exchange for Optical Asymmetric Key Cryptography, *WSEAS Transaction on Computers*, issue 2, volume 7, February 2008, pp. 52-55
[6] X6. Jiun-Jian Liaw, Yu-Sheng Liao, Lin-Huang Chang, Improvement of Blind Data Hiding Scheme in Digital Images Based on Self Refernce for Lossy Compresion, *WSEAS Transaction on Computer*, issue 5, volume 6, May 2007

[7] X7. Roshidi Din, Hanizan Shaker Hussain, Salehuddin Shuib, Hidding Secret Messages in Images: Suitability of Different Image File Types, *WSEAS Transaction on Computers*, issue 1, volume 6, January 2007
[8] X8. Youssef M.I., Emam A.E., Abd ElGhany M., Image Encryption Using PseudoRandom Number and Chaotic Sequence Generators, *Proceedings of the 7th WSEAS International Conference on INFORMATION SECURITY and PRIVACY (ISP '08)*, Cairo 2008, pp.73-78
[9] X9. Nameer N. EL-Emam, Hiding a Large Amount of Data with High Security Using Steganography Algorithm, *Journal of Computer Science*, issue 3, 2007, pp. 223-232
[10] X10. Y.-C. Lee, Y.-C. Hsieh and P.-S. You, A Secure Password Authentication Protocol for Wireless Networks, *International Journal of Computers*, Issue 3, Volume 1, 2007
[12] X12. Roshidi Din, Azman Samsudin, Digital Steganalysis: Computational Intelligence Approach, *International Journal of Computers*, Issue 1, Volume 3, 2009
[13] X13. RSA Algorithm, http://www.di-mgt.com.au/rsa_alg.html#rsasummary
[14] X14. http://inf.ucv.ro/~dan/courses/I213/ I213_CURS.pdf
[15] X15. Image Steganography and Steganalysis, http://hax.tor.hu/read/steganography/sing_stego.pdf,
[16] X16. Alain C. Brainos, A Study Of Steganography And The Art Of Hiding Information, East Carolina University, http://www.infosecwriters.com/text_resources/pdf/steganographyDTEC6823.pdf