ASPECTS OF IMAGE TRANSMISSION ON THE GOVERNEMENTAL COMMUNICATIONS SYSTEMS

CIPRIAN RĂCUCIU[(1)], CONSTANTIN DOCHIŢOIU[(1)], NICOLAE JULA[(1)], MARIAN PEARSICĂ[(2)]

[(1)]Military Technical Academy, 81- 83 George Coşbuc Blvd., 050141, Bucharest, Romania
E-mail: ciprianr@mta.ro, cdochitoiu@rdslink.ro, njula@mta.ro
[(2)]Air Force Academy, 160 Mihai Viteazul St., 2200, Braşov, Romania
E-mail: marianpearsica@yahoo.com

Abstract: Data security became a very important problem in our days. For this reason the algorithms involved get faster and more resistant to cryptographic attacks. Because of the improvement of the channels, images are more often used in data transmissions. In the present article we want to make a study regarding usage of chaotic random number generators in image cryptography and to compare the experimental results with those obtained using symmetrical algorithms. Both methods have the advantage of the speed obtained in the cryptographic process. The symmetrical algorithm used in our example is Blowfish, because it is known as a fast and a quite powerful one. We will see that the symmetrical algorithms have some disadvantages when are used in image cryptography.

Keywords: image encryption, symmetrical encryption methods, Blowfish method.

1. INTRODUCTION

1.1 Random numbers generators with random cycle lengths

The principle of random cycle lengths is exemplified by a new class of random numbers generators similar to the additive or lagged Fibonacci generators, but with extra rotation or swapping of bits. Several types are exemplified below. In a RANROT generator type A, the bits are rotated after the addition, in type B they are rotated before the addition. You may have more than two terms as in type B3 below, and you may rotate parts of the bit strings separately as in type W.

RANROT type A: $X_n = ((X_{n-j} + X_{n-k}) \bmod 2^b) \text{ rotr } r$;

RANROT type B: $X_n = ((X_{n-j} \text{ rotr } r_1) + (X_{n-k} \text{ rotr } r_2)) \bmod 2^b$;

RANROT type B3: $X_n = ((X_{n-i} \text{ rotr } r_1) + (X_{n-j} \text{ rotr } r_2) + (X_{n-k} \text{ rotr } r_3)) \bmod 2^b$;

RANROT type W: $Z_n = ((Y_{n-j} \text{ rotr } r_3) + (Y_{n-k} \text{ rotr } r_1)) \bmod 2^{b/2}$,

$Y_n = ((Z_{n-j} \text{ rotr } r_4) + (Z_{n-k} \text{ rotr } r_2)) \bmod 2^{b/2}$,

$X_n = Y_n + Z_n \cdot 2^{b/2}$.

Where Xn is an unsigned binary number of b bits, Yn and Zn are b/2 bits.

Y rotr *r* means the bits of *Y* rotated *r* places to the right (000011112 rotr 3 = 111000012). Parameters *i*, *j* and *k* are different integers. For simplicity, it is assumed that $0 < i < j < k$, and that each $r \in [0,b)$, except for type W where $r \in [0,b/2)$. The performance of these generators and the selection of good parameters are discussed below.

1.2 Image cryptography using RANROT

The idea of image cryptography using RANROT has its origin in text cryptography using random numbers, method proposed by C. Shannon. The easiest method for encrypt a character string is to create a pseudorandom numbers string and to XOR them with the character string elements, resulting the cryptogram. If when receiving the cryptogram it is known the method to generate the pseudorandom numbers, using XOR again, we will get the original string.

To generate pseudorandom numbers it is used a pseudorandom numbers generator, which in our case is RANROT. If the same initialization key is used, with the same generator, both in the encryption and in the decryption processes, the same pseudorandom numbers sequence will be generated.

The case of image encryption is similar with the one described, the only difference being that the character set is replaced by the values of the pixel color from the original picture. The rest of the process is identical.

1.3 Overview on Blowfish

Blowfish is optimized for applications where the key does not change often, like a communications link or an automatic file encryptor. It is significantly faster than DES when implemented on 32-bit microprocessors with large data caches, such as the Pentium and the PowerPC. Blowfish is not suitable for applications, such as packet switching, with frequent key changes, or as a one-way hash function. Its large memory requirement makes it infeasible for smart card applications.

Blowfish is a 64-bit block cipher with a variable-length key. The algorithm consists of two parts: key expansion and data encryption. Key expansion converts a key of up to 448 bits into several subkey arrays totaling 4168 bytes.

Data encryption consists of a simple function iterated 16 times. Each round consists of a key-dependent permutation, and a key- and data-dependent substitution. All operations are additions and XORs on 32-bit words. The only additional operations are four indexed array data lookups per round.

Blowfish uses a large number of subkeys. These keys must be precomputed before any data encryption or decryption. In total, 521 iterations are required to generate all required subkeys.

Applications can store the subkeys - there's no need to execute this derivation process multiple times.

Serge Vaudenay examined Blowfish with known S-boxes and *r* rounds; a differential attack can recover the P-array with 2^{8r+1} chosen plaintexts. For certain weak keys that generate bad S-boxes (the odds of getting them randomly are 1 in 2^{14}), the same attack requires only 2^{4r+1} chosen plaintexts to recover the P-array. With unknown S-boxes this attack can detect whether a weak key is being used, but cannot determine what it is (neither the S-boxes nor the P-array). This attack only works against reduced-round variants; it is completely ineffective against 16-round Blowfish.

2. EXPERIMENTAL RESULTS

For encryption we used a C++Builder application, which implements the Blowfish 64 algorithm (how it is described by Bruce Schneier at www.schneier.com) and the pseudorandom numbers generators RANROT type B and B3. The key used is "12345". We obtained the following results:

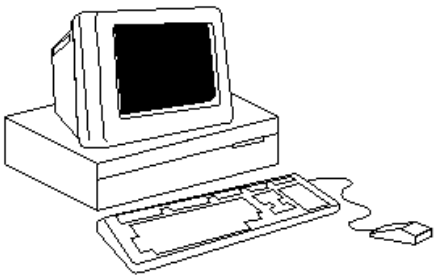

Fig. 2.1 Original (8 bits)

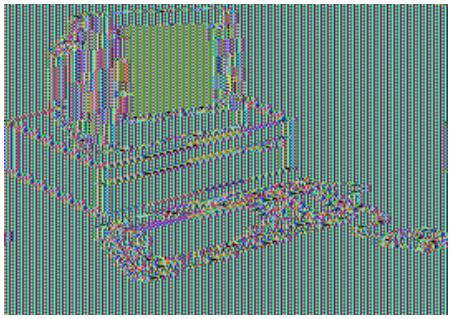

Fig. 2.2 Blowfish

Fig. 2.3 RANROT type B

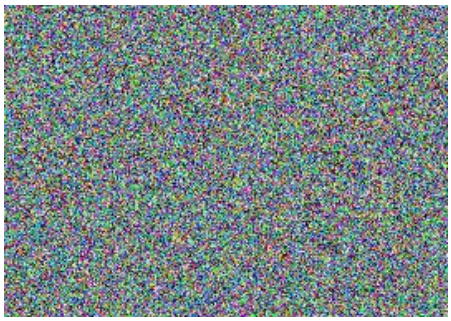

Fig. 2.4 RANROT type B3

Fig. 2.5 Original (8 bits)

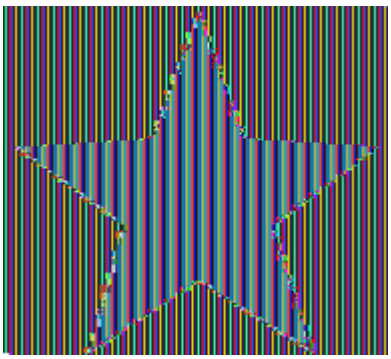

Fig. 2.6 Blowfish

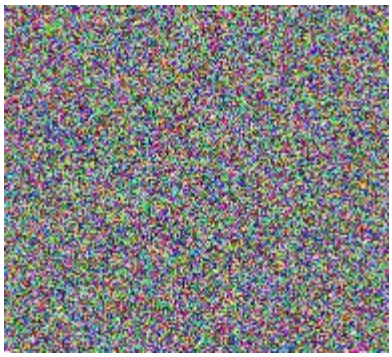

Fig. 2.7 RANROT type B

Fig. 2.8 RANROT type B3

Fig. 2.9 Original (32 bits)

Fig. 2.10 Blowfish

Fig. 2.11 RANROT type B

Fig. 2.12 RANROT type B3

3. CONCLUSIONS

From the experimental tests we can observe the advantage of pseudorandom numbers systems. All the images have been encrypted using 32 bits blocks and 64 for Blowfish. Because Blowfish does in fact a substitution, replacing 64 bits blocks with others, it proves serious disadvantages regarding simple images, with a dominant color, those being recognized if the cryptogram is interpreted like image (the same size like original). However, blowfish prove to be good if the image is more complex and in high definition. This disadvantage is overcome if pseudorandom numbers are used.

REFERENCES:

[1] Couture, R and L'Ecuyer, «Distribution Properties of Multiply-with-Carry Random Number Generators». *Mathematics of Computation*, vol. 66, pg. 591, P. 1997.
[2] Couture, R and L'Ecuyer, «Guest Editors' Introduction». *ACM transactions on Modeling and Computer Simulation*. vol. 8, no. 1, pg. 1-2, P. 1998.
[3] Fog, A., 2001. « Pseudorandom number generators», http://www.agner.org/random.
[4] Răcuciu, C., «Contributions Related to the Improvement of the Image Encryption Methods», PhD Thesis, Bucharest, 2003.
[5] *** Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish) – http://www.schneier.com