Judgment of Extracting Encryption Keys From Image Data

Omer Abu Shqeer

College of Computer Science and Engineering, Taibah University, KSA

Summary
Huge amount of sensitive data transmit over networks every day, cryptography is famous and widely used technique to prevent these data from unauthorized people; encryption/decryption strength depends on the algorithm and the key(s). Keys management including generation, goodness and distribution are very important and of the concern of the researchers. This paper focused on the keys generation from images data and highlighted the good features of the image that can be used for keys generation. A proposed algorithm for keys generation from images has been developed, three randomness tests: frequency/mono-bit, serial and poker tests have been implemented. The algorithm has been used to generate large number of keys of different lengths (256-bits and 512-bits) from variety of images with different features, the generated keys have been evaluated for randomness quality, results have been classified according to images features (entropy value and number of colors) and then analyzed and concluded. By conclusion, images data are not a good choice for keys generation, but some images that have a large number of colors and big entropy value can be used for this purpose.

Keywords:
encryption, key, image, entropy, randomness.

1. Introduction

Sensitive and confidential data is very valuable and therefore should be stored and transmitted securely. Cryptography and steganography are used to do so. Steganography hides the subject data into another data, so it becomes hidden to hackers and intruders. Cryptography transform the data into unreadable form using encryption algorithm and a key, so even if it is available, no one can understand it unless it is decrypted using the correct decryption algorithm and the appropriate key. Encryption can be symmetric (using same key for both encryption and decryption) and asymmetric (using different keys for encryption and decryption). The security of encrypted data entirely depends on two things: the strength of the cryptographic algorithm and the secrecy of the key. Strong key should be uniformly distributed and precisely reproduced. Regardless of the type of encryption, the encrypted data is only secure if the encryption key is protected. The best passphrases are alphanumeric and random, though these are harder to remember [1][2][3].

Many techniques are used to generate the encryption and decryption keys. Some of these techniques may use image data to generate the required keys if this data satisfies the randomness requirements. Hence, a suitable metrics are needed to investigate the degree of randomness of the key bits sequence [4]. In this paper I introduced some statistical tests used in evaluation of randomness and the approaches used to do so. I proposed a technique to generate keys from image data, and used the mentioned statistical tests to evaluate the randomness of the generated keys from image data, and finally I set guidelines for the suitable images that can be used to generate encryption keys based on the image entropy and tests results.

2. Image entropy

One of the image attributes that are useful in image processing is the image entropy. Entropy is a measure that quantifies the information contained in an image. An image of little contrast and large runs of pixels with the same or similar color values has low entropy. On the other hand, an image with a great deal of contrast from one pixel to the next has high entropy (i.e. a contrast-rich image has a high entropy value) [5][6].

A good estimate of entropy is usually not available, but the following are two methods that are used to calculate the entropy. The first has been given by Galileo Imaging Team as shown in equation 1.

$$E = -\sum_{i} P_i Log_2 P_i \quad (1)$$

Where:
E : Entropy Value
Pi : The probability that the difference between two adjacent pixels is equal to i
Log 2: The base 2 logarithm

The second has been given by equation 2 below.

$$H_e = -\sum_{k=0}^{G-1} P(k) \log_2 (P(k)) \quad (2)$$

Where:
He : Entropy value
G : Gray levels that the image has
P(k) : The probability of gray level k.

Manuscript received February 5, 2014
Manuscript revised February 20, 2014

Image entropy can however be estimated from a gray level histogram as follows: Let an image size be W×H, number of gray levels is 2b where b is the bits-depth, and f(k) be the frequency of gray level k in the image where; 0 <= k <= 2b –1, then the estimated probability occurrences of gray level k is given in equation 3 and the entropy is given in equation 4.

$$\tilde{P}(k) = \frac{f(k)}{w \times h} \quad (3)$$

$$\tilde{H}_e = -\sum_{k=0}^{2^b-1} \tilde{P}(k) \log_2(\tilde{P}(k)) \quad (4)$$

Where:
He : Entropy level
P(k) : The estimated probability occurrences of gray level k
b : Log2 (Number of gray levels)

3. Encryption key

Even if we have a strong encryption algorithm, we must also generate strong keys so that the security of the data isn't undermined by weak cryptographic keys. Key length is one of the measurement factors of the key strength but is not enough. The data used to generate the key and the key itself must be sufficiently random as determined by the data security requirements [7]. Many algorithms and sources are used to generate encryption keys including random number generators, natural data with random property and fuzzy extraction from biometrics and noisy data [8]. In some cases, images data can be used to generate the encryption keys. This section presents a proposed simple algorithm to generate keys from images in order to be used in my experiments, and then some statistical tests are used to evaluate the randomness level of the generated keys.

3.1.Key selection/generation

Various approaches can be used to generate a key from an image data. In this proposal, I assumed that the key is selected from an image data one bit at a time in a random fashion until the required number of bits chosen. In this case the number of possible keys (NoPK) with length k from an image data of n bits, with assumption that any bit can be selected more than once is given by equation 5.

$$NoPK = n^k \quad (5)$$

For example: Consider a 640 × 480 image with 24-bits resolution and a key of length 512 bits, then we have (640 × 480 × 24)512 possible keys. Following is the proposed algorithm.

ALGORITM Key_Selection
Load image
Input k /* k: key length */
n = number of image pixels * bits resolution/* n: # of bits in the image */
Convert image data into ImageBitsArray(n)
For i = 0 to k-1
r = random integer between (0, n-1) /* r: a bit position in the image */
Key_array(i) = ImageBitsArray(r)
Next i
END Key_Selection
Input: An image, key length
Output: A key of length k.

3.1.Evaluation of key randomness

A lot of options are available to analyze cryptographic RNG. Among these are The National Institute of Standards and Technology (NIST) statistical test suite, and the tests described by Donald Knuth in his book 'the art of computer programming, semi-numerical algorithms, volume 2'. The most commonly used tests are mono-bit or frequency, frequency test within a block, runs, longest runs of ones, serial, poker, and autocorrelation. Each test aims to test such characteristic(s) of the random sequence. Different evaluation approaches: threshold value, fixed range, and probability value are used to decide whether a sequence passed or failed a statistical test. In this paper I explained only three statistical tests as a sample to be used in my experiments, these are: mono-bit, serial and poker tests. I used only those tests because my goal is to check the effect of image entropy and number of colors on selecting an image for key generation. [9][10][11].

3.1.1.Frequency/Mono-bit test

"The purpose of this test is to determine whether the number of ones and zeros in a sequence are approximately the same as would be expected for a truly random sequence. The test assesses the closeness of the fraction of ones to ½" [11]. The test statistic is given by equation 6, and the P-value by equation 7 [11].

$$S_{obs} = \frac{|S_n|}{\sqrt{n}} \quad (6)$$

Where;
n: Sequence length
$S_n = X_1 + X_2 + \ldots + X_n$, where, $X_i = 2\varepsilon_i - 1$
ε: i^{th} input of the sequence

$$P\_Value = erfc(\frac{S_{obs}}{\sqrt{2}}) \quad (7)$$

3.1.1.Serial Test

The purpose of this test is to determine whether the number of occurrences of m-bit overlapping patterns is approximately the same as would be expected for a random sequence [11]. With the assumption that m=2, this test can be accomplished as given in equation 8.

$$STV = \frac{4}{n-1}(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2) - (n_0^2 + n_1^2) + 1 \quad (8)$$

Where;
STV: Serial Test Value
n0: Number of 0's
n1: Number of 1's
n00: Number of occurrences of the pattern 00
n01: Number of occurrences of the pattern 01
n10: Number of occurrences of the pattern 10
n11: Number of occurrences of the pattern 11
n: The sequence length

3.1.1.Poker Test

The purpose of this test is to determine whether the number of ones and zeros in each of m non-overlapping blocks created from a sequence appear to have a random distribution [11]. This test can be accomplished as given in equation 9.

$$PTV = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k \quad (9)$$

Where;
PTV: Poker Test Value
m : Length of each pattern in bits
k : Number of non-overlapping patterns
ni : Number of occurrences of the ith pattern

4. Experiments and Results

We applied the tests measurements that mentioned previously on different images categorized in low, medium, and high entropy values. Within each category I chose different images such that some of them have unbalanced number of 1's and 0's in the image data bits, and some other have almost balanced number of 1's and 0's in the image data bits. Hundred thousands keys of length 256 and another hundred thousands keys of length 512 were generated from each image. All the images are of size 400 × 300 pixels and of 8-bits depth. Figure-1 below shows the used images, table-1 presents detailed results for each of the randomness statistical tests for every image, and table-2 presents cumulative results of the randomness statistical tests for every image. Randomness tests have been executed with the following assumptions:
The significance level ($\alpha = 0.05$).
In serial test, m is assumed to be 2 bits (i.e. patterns are 00, 01, 10, and 11).
In poker test, m is assumed to be 12 blocks; each of them has 80000 bits.
It is clear from tables 1 and 2 that images of zero entropy value (i.e. images from 1 to 4) are not suitable at all where the percent of keys that passed the randomness tests is almost zero; from this point and forward I will ignore these extreme images from my discussion. Table 3 presents cumulative results as well as their averages of the randomness tests for the images of entropy value greater than zero and the generated keys of length 256-bits; while table-4 presents these results for the 512-bits keys. It is clear from table-3 that only 12.6% of the generated keys of length 256-bits failed to pass any randomness test, and this percent is bigger with 512-bits keys where it is about 23% as shown in table-4; Also it is clear from the tables that 52.2% of the generated keys of length 256-bits passed all the three randomness tests, and less percent 42.1% passed all the three randomness tests for 512-bits keys. The data in tables 3 and 4 also shows that images of higher entropy values almost give better randomness tests results. The column charts in figures 2 and 3 summarize the data of those tables respectively.

Let us now discuss the effect of the image colors on the randomness tests results. We know that every color in a digital image is represented by a sequence of bits, and therefore any digital image is a sequence of 1's and 0's. Tables 5 and 6 present cumulative results of the randomness tests for the images of entropy value greater than zero and the generated keys of lengths 256-bits and 512-bits respectively, data in those tables are sorted according to the percent of 1's in the image bits. It is very clear that images with balanced number of 1's and 0's in the image data (i.e. percent of 1's is almost 50%) satisfied better randomness results for the generated keys of both lengths. The column charts in figures 4 and 5 summarize the data of tables 5 and 6 respectively.

5. Conclusion

This paper presents the importance of the encryption/decryption keys in the field of data security. Sources of these keys, generating algorithms and goodness are important issues. In this research I proposed a simple algorithm to generate the keys from images data, tested that algorithm on variety of images and evaluated the randomness quality of the generated keys, and discussed the effect of image entropy value and image colors on the randomness quality. Through the analysis of the experiments results I can conclude the following:

Even though some images of the experiments showed good results and some other showed bad results we can say that image data may be used to generate encryption keys, but at the same time we can say that images data are not a perfect choice to be used for keys generation.

Images with very low entropy value (i.e. they have a limited number of colors) are not suitable for keys generation; while images with high entropy value (i.e. they have variety of colors) can be used to generate keys.

Images with unbalanced number of 1's and 0's in the image data bits are not suitable for keys generation; while images with balanced number of 1's and 0's in the image data bits are good to be used.

The best images to be used are those that have bigger entropy value with balanced number of 1's and 0's in the image data bits.

Finally, this conclusion is based on the three mentioned randomness tests and can be enhanced by applying more randomness tests on the experimental images.

References

[1] Pfleeger, C. P. and Pfleeger, S. L. (2007). Security in Computing, 4th ed. Prentice Hall.

[2] Stallings, W. (2010). Cryptography and network security – principles and practices, 5th ed. New Jersy, USA. Prentice Hall.

[3] Kurose J and Ross K. (2010). Computer networking, 5th ed. Addison Wesley.

[4] Juan Soto, Statistical testing of random number generators, National Institute of Standards & Technology.

[5] Mohammad Ali Bani Younes and Amman Jantan (2008), An image encryption approach using a combination of permutation technique followed by encryption, IJCSNS vol.8, No.4.

[6] Jayant Kushwaha and Bhola Nath Ro, Secure Image Data by Double encryption, International Journal of Computer Applications (0975 – 8887) Volume 5– No.10.

[7] https://www.owasp.org/index.php/Cryptographic_Storage_ Cheat_Sheet, cited on 9/2/2013.

[8] YevgeniyDodis et al (2008). Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data, SIAMJournalonComputing, 38(1):97-139.

[9] S. KIM, K. UMENO, A. HASEGAWA, (2003). On the NIST statistical test suite for randomness ,IEICE, Technical Report,Vol.103, No.449, pp21-27.

[10] NIST (2001), FIPS PUB 140-2, Security Requirements for Cryptographic Modules.

[11] (http://www.us.designeuse.com/exit?url=http://csrc.nist.gov /publications/fips/fips140-2/fips1402.pdf).

[12] NIST (2001), Special Publication 800-22, A statistical test suite for random and pseudo-random number generators for cryptographic applications. (http://csrc.nist.gov/rng/)

Omer Abu Shqeer: received the B.Sc., M.Sc, and PhD degrees in computer science from Yarmouk univ. – Jordan (1986), Univ. of Sains Malysia – Malysia (2002), and Amman Arab Univ. for Graduate Studies – Jordan (2006) respectively. He was a programmer and systems analyst in Jordan universities (1990 – 1996), Lecturer in computing in Oman technical colleges (1996 – 2001), Lecturer in Philadelphia univ. – Jordan (2002 – 2006) and then assistant prof. (2006 – 2011), and now assistant prof. in Taibah univ. – KSA (2011 – Now).