Journal of Computer Science 4 (12): 999-1002, 2008
ISSN 1549-3636

A Shuffle Image-Encryption Algorithm

Abdelfatah A. Yahya and Ayman M. Abdalla
Department of Computer Science, Al-Zaytoonah University of Jordan,
Amman 11733, Jordan

Abstract: Problem statement: Image encryption needs to be secure by resisting statistical attacks and other types of attacks. **Approach:** The new algorithm, call it the Shuffle Encryption Algorithm (SEA), applies nonlinear s-box byte substitution. Then, it performed a shuffling operation partially dependent on the input data and uses the given key. **Results:** SEA was implemented and tested with different data, mainly consisting of images. Results confirmed its security, shown through statistical analysis using histograms, correlation and covariance. **Conclusion:** New algorithm was suited for encrypting images and other types of data.

Key words: Cryptography, image encryption, s-box, rijndael

INTRODUCTION

The study of cryptography has become increasingly important, as progressively more information is stored and transmitted in electronic form. This led to the creation of many encryption algorithms and standards. One of the most popular standards today is the Advanced Encryption Standard (AES), also known as Rijndael. This standard is the cipher of choice for many official and commercial organizations around the world and it is used in encrypting many forms of electronic data.

A detailed explanation of the design of AES is available[5]. The AES uses byte substitution using a table called an s-box. More s-box construction methods were developed later[1,4]. AES may also be implemented efficiently on smart cards[9]. Some studies were made on image encryption using AES[2,7] and using matrix transformation[10,11]. The security of AES was demonstrated by its resistance to attacks when it was applied with ten or more rounds using a key of at least 128 bits[3,8,12-14]. Other encryption algorithms based on AES were also developed[6,12,13].

A new encryption algorithm is presented. The new algorithm, called the Shuffle Encryption Algorithm (SEA), uses a nonlinear byte-substitution step using an s-box and a shuffle operation. This new algorithm is reasonably secure, as shown by the implementation results.

MATERIALS AND METHODS

The new SEA algorithm starts with nonlinear byte substitution using a lookup table (s-box) and then it performs linear shuffling of the result. These two steps are performed for k rounds, where the key consists of k numbers, k≥1. The shuffling step is both key dependent and data dependent.

Let the bytes of an (n×m)-byte image be numbered left to right, row by row, starting with number (j = 1) for the first byte of the first row and ending with number (j = nm) for the last byte of the last row. The encryption algorithm using SEA is as follows:

```
Generate s-box
For i = 1 to k
    fixBit = Key[i]
    D  = Vector where D[j] is the value of bit (fixBit)
         of the j^th byte of the current image
    S0 = Vector containing numbers of current image
         bytes (j) that have (D[j] = = 0)
    S1 = Vector containing numbers of current image
         bytes (j) that have (D[j] = = 1)
    Shuffle = Concatenation of S0 with S1
    Substitute the bytes of the current image so that the
    new location of byte (j) is byte (Shuffle[j])
    Substitute the bytes of the resulting image using the
    s-box table
End For
```

This algorithm takes an image (or another data item) and a key consisting of k numbers as input and it works as follows. First, an s-box substitution table is constructed to perform two transformations: multiplicative inverse and affine transformation. The algorithm takes k rounds, where k≥1. Each round takes

Corresponding Author: Abdelfatah A. Yahya, Faculty of Science and Information Technology,
Al-Zaytoonah University of Jordan, Queen Alia Airport Road, Amman 11733, Jordan

one number from the key to specify the bit location (fixBit) in each byte, where 0≤fixBit≤7. Then, a shuffle vector is constructed by listing the numbers of bytes with the value of bit number fixBit equal to zero, followed by the numbers of bytes with the value of bit number fixBit equal to one. This vector gives a mapping that specifies the new location of each byte in the image. Finally, a different byte substitution is performed using the s-box table.

Each round, after the first, uses a different fixBit and applies the same steps to the image that resulted from the preceding round. Note that the value of fixBit can be represented by three bits, which makes the length of the key 3k bits.

Encryption using SEA is applied in this simple example. Let the binary representation of the input be: 1111<u>0</u>101 0010<u>1</u>101 1010<u>0</u>011 1000<u>1</u>100 with key = (3, 5). This key consists of two numbers, so two iterations should be applied. In the first iteration, fixBit = 3 and this bit is underlined in the input above. Based on the values of this fixBit, S0 = (1, 3) and S1 = (2, 4), which make the shuffle vector (1, 3, 2, 4). Then, the input after the shuffle substitution will be 11110101 10100011 00101101 10001100. After that, the s-box substitution is applied, where each of these byte values is replaced with its lookup value in the s-box. For this example, suppose the result is 11<u>1</u>00110 00<u>0</u>01010 11<u>0</u>11000 01<u>1</u>00100. This result is used by the second iteration, where fixBit = 5 and these bits are underlined. Now, S0 = (2, 3) and S1 = (1, 4), which make the shuffle vector (2, 3, 1, 4) and its result is 00001010 11011000 11100110 01100100. Finally, s-box substitution gives the result 011000111 011000001 10001110 0100001.

The decryption algorithm is similar to the encryption algorithm, but with replacing the shuffle operation with its inverse and using the inverse s-box substitution at the beginning of the iteration. This decryption restores the original image without loss of quality. The decryption algorithm is described:

Generate inverse s-box
For i = k to 1 step -1
    fixBit = Key[i]
    Substitute the bytes of the current image using the inverse s-box table
    D = Vector where D[j] is the value of bit (fixBit) of the j^{th} byte of the resulting image
    S0 = Vector containing numbers of image bytes (j) that have (D[j] = = 0)
    S1 = Vector containing numbers of image bytes (j) that have (D[j] = = 1)
    Shuffle = Concatenation of S0 with S1
    Substitute the bytes of the image so that the new location of byte number (Shuffle[j]) is byte number (j)
End For

RESULTS

SEA was implemented with MATLAB programming language since it is known for its readability and strength in the manipulation of matrices and images. The implementation was done on a personal computer with a 2.2 GHz Core2Duo processor and one-gigabyte main memory, running with the Windows XP operating system. SEA was applied to 100 colored images of various types, with sizes ranging from three to 400 kilobytes.

The histograms of encrypted images appeared uniform and different from the histograms of the original images. Consider the sample picture (Zaid) shown in Fig. 1 and its histogram in Fig. 2. As shown in Fig. 3 and 4, the histogram of the encoded image gives no indication that may help a statistical attack, even after a single iteration of the algorithm.

Fig. 1: The original Zaid image

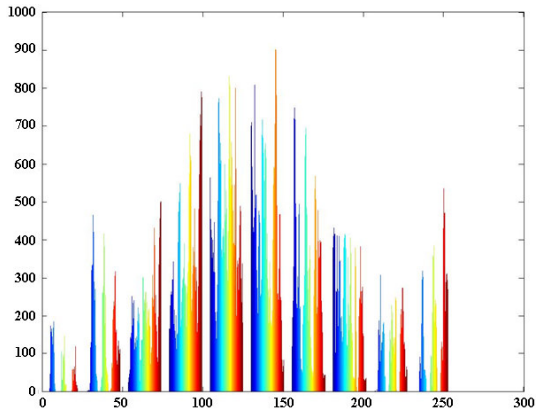

Fig. 2: Histogram of the original Zaid image

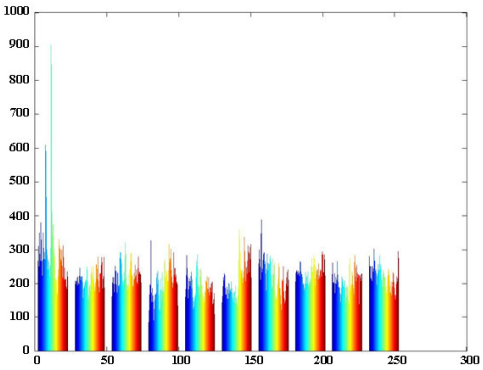

Fig. 3: Histogram of the encoded image after one iteration

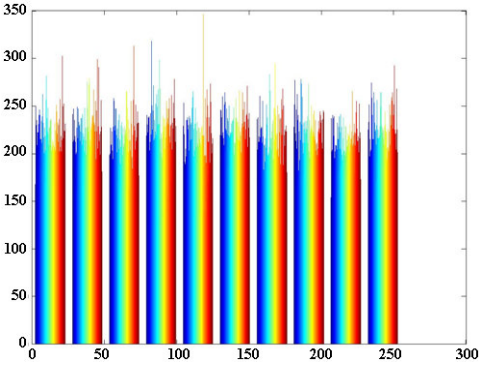

Fig. 4: Histogram of the encoded image after ten iterations

Let X be the linear form of an image. Let E(y) be the expected value for variable y. Then, the covariance matrix for an image is computed using the following equation:

$$C[i, j] = E((X[i] - E(X[i])(X[j] - E(X[j]))$$

For example, the covariance matrices of the sample original image and its encoded image (after 10 iterations) are plotted in Fig. 5 and 6, respectively. When the matrices of Fig. 5 and 6 were compared, there was no resemblance between them, which makes a statistical attack very difficult.

The correlation, r, between two images, stored in matrices A and B, is computed as follows, where, $\bar{A}$ and $\bar{B}$ are mean values for matrices A and B, respectively:

$$r = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(A[i,j]-\bar{A})(B[i,j]-\bar{B})}{\sqrt{\left(\sum_{i=1}^{m}\sum_{j=1}^{n}(A[i,j]-\bar{A})^2\right)\left(\sum_{i=1}^{m}\sum_{j=1}^{n}(B[i,j]-\bar{B})^2\right)}}$$

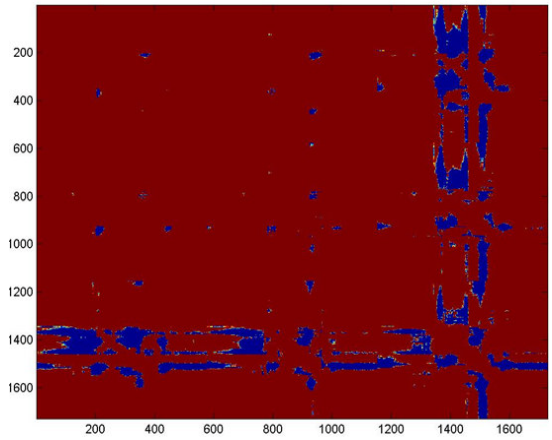

Fig. 5: Covariance for the original image

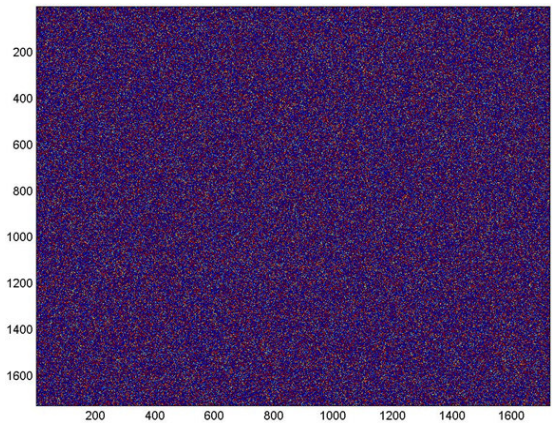

Fig. 6: Covariance for the encoded image

The correlation between the sample original image and the encrypted image was -5.98×10^{-4} after one iteration and 2.15×10^{-4} after ten iterations of the encryption algorithm. This shows there is little resemblance between the original image and the encrypted image, even with a single iteration. In addition, the encryption was sensitive to the fixBit. For the selected sample image, the correlation between the images encrypted with one iteration using fixBit = 6 and fixBit = 7 was -12×10^{-4}. The correlation values computed for other images gave similar results.

DISCUSSION

The security of SEA comes from the shuffle operation and the nonlinear s-box byte substitution. If one or more bits in the key are changed, a different shuffle bit is chosen and the substitution is changed. There are $k\times2^b$ different possible shuffle vectors for an

input of size b bytes encrypted in k iterations. In addition, there are 2,048 different s-boxes of size 16×16 with 8 bits for each entry. This makes the total number of byte permutations at least 2^{b+11}. Consequently, for an input item of one or more kilobytes, a brute-force attack is impossible.

When the same key was used for encrypting different images using SEA, it generated different shuffle vectors based on the values of selected bits in the new image, which resulted in producing different encrypted images. When different keys were used with the same image, they produced different encrypted images. In addition, analysis using histograms, covariance and correlation show properties of SEA that strongly resist statistical attacks.

The security properties demonstrated by the above analysis imply the algorithm's security in general. However, other types of attacks, not related to statistical or brute-force attacks, are possible. The SEA algorithm may be combined with an algorithm that changes the contents of the cache by not allowing any data items to remain in the cache for too long. Such algorithms succeeded in preventing cache timing attacks on AES and other algorithms, and therefore they can help SEA. Other types of attacks on SEA may also be studied in future research.

CONCLUSION

A new encryption algorithm was presented. The algorithm, SEA, uses the s-box non-linear byte substitution used by other cipher algorithms, such as AES. Then, it applies a linear byte-shuffling operation. Statistical analysis using histograms, correlation and covariance showed SEA is not vulnerable to statistical attacks. In addition, the huge number of possible keys makes a brute-force attack on SEA impossible.

REFERENCES

1. Abuelyman, E.S. and M.A. El-Affendi, 2007. A real time s-box construction using arithmetic modulo prime numbers. J. Digital Inform. Manage., 5: 354-260. http://www.dirf.org/jdim/v5i60a2.pdf
2. Benabdellah, M., M.M. Himmi, N. Zahid, F. Regragui and E.H. Bouyakhf, 2007. Encryption-compression of images based on FMT and AES algorithm. Applied Math. Sci., 1: 2203-2219. http://www.m-hikari.com/ams/ams-password-2007/ams-password45-48-2007/benabdellahAMS45-48-2007.pdf
3. Biham, E., O. Dunkelman and N. Keller, 2006. Related-key impossible differential attacks on 8-round AES-192. Lecture Notes Comput. Sci., 3860: 21-33. http://www.springerlink.com/content/p804p313030l1680/
4. Chen, G., 2008. A novel heuristic method for obtaining S-boxes. Chaos Solit. Fract., 36: 1028-1036. DOI: 10.1016/j.chaos.2006.08.003
5. Federal Information Processing Standards (FIPS 197), 2001. The Advanced Encryption Standard. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
6. Duc, D.A., T.M. Triet and L.H. Co, 2002. The extended rijndael-like block ciphers. Proceeding of the International Conference on Information Technology, Coding and Computing, Apr. 8-10, IEEE Xpolre, USA., pp: 183-188. DOI: 10.1109/ITCC.2002.1000384
7. El-Fishawy, N. and O.M. Abu Zaid, 2007. Quality of encryption measurement of bitmap images with RC6, MRC6 and rijndael block cipher algorithms. Int. J. Network Secur., 5: 241-251. http://cmpe.emu.edu.tr/chefranov/cmpe553_07_Spring/Notes/ijns-2007-v5-n3-p241-251.pdf
8. Jakimoski, G. and Y. Desmedt, 2004. Related-key differential cryptanalysis of 192-bit key AES variants. Lecture Notes Comput. Sci., 3006: 208-221. http://www.springerlink.com/content/8gnbt35440jx683y/
9. Lu, C.F., Y.S. Kao, H.L. Chiang and C.H. Yang, 2004. Fast implementation of AES cryptographic algorithms in smart cards. Proceeding of the IEEE 37th Annual International Carnahan Conference on Security Technololgy, Oct. 14-16, IEEE Xpolre, USA., pp: 573-579. DOI: 10.1109/CCST.2003.1297622
10. Shuangyuan, Y., L. Zhengding and H. Shuihua, 2004. An asymmetric image encryption based on matrix transformation. Proceeding of the IEEE International Symposium on Communications and Information Technology, Oct. 26-29, IEEE Xpolre, USA., pp: 66-69. DOI: 10.1109/ISCIT.2004.1412451
11. Shuihua, H. and Y. Shuangyuan, 2005. An asymmetric image encryption based on matrix transformation. ECTI Trans. Comp. Inform. Technol., 1: 126-133. http://www.ecti-thailand.org/PDF/Transaction/CIT_1_2/P_02.pdf
12. Zeghid, M., M. Machhout, L. Khriji, A. Baganne and R. Tourki, 2007. A modified AES based algorithm for image encryption. Int. J. Comp. Sci. Eng., 1: 70-75. http://www.waset.org/ijcse/v1/v1-1-9.pdf
13. Zeghid, M., M. Machhout, L. Khriji, A. Baganne and R. Tourki, 2007. A modified AES based algorithm for image encryption. Proceeding of the World Academy of Science, Engineering and Technololgy, May, WASET Organization, USA., pp: 206-211. http://www.waset.org/pwaset/v21/v21-37.pdf
14. Zhang, W.T., W.L. Wu and L. Zhang, 2007. Related-key impossible differential attacks on reduced-round AES-256. J. Software, 18: 2893-2901. http://www.lois.cn/LOIS-AES/data/AES-256.pdf