Neuro-Cryptanalysis of DES and Triple-DES

Mohammed M. Alani

Department of Computing, Middle-East College, Muscat, Sultanate of Oman.

m.alani@d-crypt.org

Abstract. In this paper, we apply a new cryptanalytic attack on DES and Triple-DES. The implemented attack is a known-plaintext attack based on neural networks. In this attack we trained a neural network to retrieve plaintext from ciphertext without retrieving the key used in encryption.The attack was practically, and successfully, implemented on DES and Triple-DES. This attack required an average of 211 plaintext-ciphertext pairs to perform cryptanalysis of DES in an average duration of 51 minutes. For the cryptanalysis of Triple-DES, an average of only 212 plaintext-ciphertext pairs was required in an average duration of 72 minutes. As compared to other attacks, this attack is an improvement in terms of number of known-plaintexts required, as well as the time required to perform the complete attack.

Keywords : cryptanalysis, des, triple-des, 3des, neural, neuro-cryptanalysis

1 Introduction

Cryptanalysis of block ciphers has witnessed rapid developments in the last few years. Many new cryptanalytic techniques were developed, especially after the adoption of the Advanced Encryption Standard (AES). Most of the recent developments in cryptanalysis focused on algebraic aspects of block ciphers [1].

In general, attacks can be classified, according to the assumed capabilities of an adversary, to the following; ciphertext-only, known-plaintext, chosen-plaintext, and chosen-ciphertext attacks. In this paper we will focus on known-plaintext attacks which assume that an adversary knows some plaintexts-ciphertexts pairs. The objective of this attack is to find the associated key or to recover unknown parts of the plaintext.

An attack may not only provide information about the key but also about other kinds of data. According to the type of information recovered, the possible outcomes of an attack can be classified as in [2]:

1. Total Break: an attacker finds or reconstructs the secret key.
2. Global Deduction: an attacker finds an algorithm functionally equivalent to the original encryption/decryption ones but without knowing the secret key.
3. Instance or Local Deduction: an attacker finds the plaintext or ciphertext of an intercepted ciphertext or plaintext which was not obtained from the legitimate parties.

adfa, p. 1, 2011.

4. Information Deduction: an attacker obtains some information about the key, plaintext or ciphertext which did not come from the legitimate parties and which was not known before the attack.

This paper uses a known-plaintext attack that is aimed to lead to global deduction of the DES and Triple-DES ciphers.

DES falls in a group of ciphers categorized as Feistel Ciphers [3]. It is built on the idea of dividing the plaintext bits into two groups, one of which is passed into an encryption function, while the other is held until the function is done. Then, the two parts are interchanged and the cycle repeated for a number of rounds.

DES has 16 rounds in which it uses sixteen 48-bit subkeys generated from an original 56-bit key [4]. Most of the DES security lies in its S-boxes which were carefully designed to resist attacks. These S-boxes are considered the only non-linear operation in the DES.

DES was adopted as a standard in 1977 [4]. Since then, this cipher was used in many applications worldwide. The high processing ability rendered the 56-bit key of the DES vulnerable to brute-force attacks. This attack, which is basically trying every possible key, was confronted by introducing the Triple-DES by the National Institute of Standards and Technology (NIST) as a more appropriate alternative algorithm, offering a higher level of security [5]. The complete description of DES can be found in [4].

Triple-DES is a repetition of DES algorithm for three times using three different keys, which sums the total key to be 168 bits, in a pattern called EDE. EDE is simply the process of Encryption-Decryption-Encryption. Each one of these processes is conducted using a different 56-bit key [6].

1.1 Cryptanalysis of DES and Triple DES

The DES has been a target for many attacks for a long time. Some of these attacks started by analyzing reduced-rounds DES and went up to the full-round DES. The most well known two attacks were, differential cryptanalysis, and linear cryptanalysis [5][7].

Differential cryptanalysis is a chosen-plaintext technique developed by Biham and Shamir in [5] against reduced-round variants of the DES cipher, and later applied to the full 16-round DES [8]. Differential cryptanalysis analyzes the effect of differences of pairs of plaintext blocks on the distribution of differences of the ciphertext pair. For DES, the difference operation was defined as bitwise exclusive-or. In general, the difference operator is selected in order to make the text difference independent of the key, which is assumed to be fixed. The full-round DES was said to be broken by differential cryptanalysis using 2^{47} chosen-plaintexts with a time complexity of 2^{37}. Many developments were introduced to this technique, one of which was impossible-differentials introduced by Biham in [9].

Linear cryptanalysis is a statistical, known-plaintext attack on block ciphers. This technique has been more extensively developed by Matsui in attacks on the DES cipher in [7]. To obtain linear relations covering the full cipher, or most of it, a general approach is to start with local approximations to non-linear components in the cipher.

In DES the S-boxes are the only non-linear transformations. All the remaining components, such as the data expansion, and the permutations are linear transformations. The full linear cryptanalysis of DES required 2^{43} known-plaintexts. Many other cryptanalytic attacks were developed afterwards, such as differential-linear cryptanalysis [10], related-key attacks [11][12], and non-linear cryptanalysis [13].

Triple-DES was a target for a slightly different type of attacks due to its repetition of DES algorithm. Repeated encryption was bounded by some limitations as mentioned in [14] and [15]. Meet-in-the-middle attacks rendered the effective key to be of 112-bits length [16]. Few attacks on Triple-DES were successful in terms of numbers of known/chosen ciphertexts/plaintexts needed to break the algorithm such as the attack introduced by Lucks which requires around 2^{32} known plaintexts, 2^{113} steps, 2^{90} single DES encryptions, and 2^{88} memory. Although it is the most successful attack on Triple-DES in terms of number of known-plaintexts needed, it was, and still, considered impractical by the NIST [17]. Few other attacks on Triple-DES reduced the number of trials needed to perform brute-force attack. In 1996, key-schedule attacks were introduced to Triple-DES among other block ciphers [18]. In 1999 parallel collision search was introduced by [19]. In 2005, another approach, relying on related-key and meet-in-the-middle attacks, was published [20]. However, many commercial applications still use Triple-DES as their encryption choice such as financial transactions, and email security.

1.2 Neural Networks and Cryptology

There have been few attempts to use neural networks in cryptography. In 1998, Clark and Blank introduced a cryptographic system based on neural networks [21]. A close encounter was made in [22] in 2005. Another application of neural networks in cryptography was published by Kinzel and Kanter which introduced a way of using neural networks in secret key exchange over a public channel [23]. Klimov, Mityagin, and Shamir introduced a method of cryptanalysis to the previous system in [24]. A scheme for remote password authentication was published in [25]. Another important development in the neural networks and cryptography interaction was the work done in [26] which suggested using neural networks in optimizing differential and linear cryptanalysis. On the other hand, a genetic-algorithm-based attack was suggested on neural cryptography in [27]. Li, in [28], suggested a new cryptographic system which was later analyzed by [29].

Recently, [30] suggested the use of right-sigmoidal function as an activation function in a neural network used in cryptanalysis of Feistel-type ciphers. This paper did not achieve actual cryptanalytic results and contained many assumptions.

Another use of neural networks in cryptanalysis was suggested by Chandra and Varghese in [31]. This paper employed neural networks in the classification of ciphertext according to the encryption algorithm used to produce it.

2 Neuro-Cryptanalysis

The attack implemented in this paper is a known-plaintext attack. This attack is based on training a neural network to do the decryption process without knowing the key. This attack was first introduced in [32] and was applied to DES. A schematic diagram of the proposed attack is shown in figure 1.

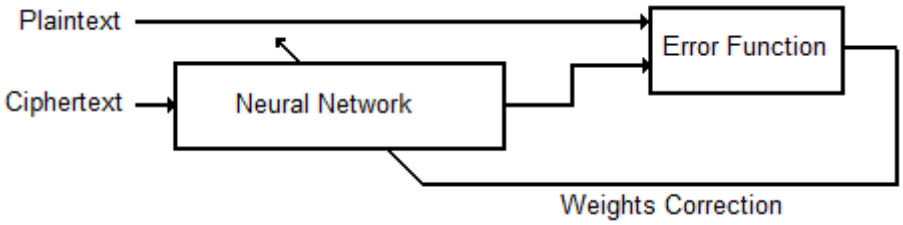

Fig. 1. A Schematic Diagram of the Neuro-Cryptanalysis System [32]

The neural network is fed with ciphertext as its input, and the plaintext is considered the reference output. After an adequate amount of training, with an adequate amount of plaintext-ciphertext pairs that are all encrypted with the same key, the neural network will be able to retrieve plaintext from ciphertext that has not been part of the training process, as long as this ciphertext is encrypted with the same key. As figure 2 shows, the key bits are not retrieved. Thus, the attack is considered a global deduction attack which was defined earlier as finding an algorithm functionally equivalent to the original decryption one but without knowing the secret key. Since the non-linearity is an essential part of every block cipher, the number of training cycles as well as the number of plaintext-ciphertext pairs is expected to be high. To have a neural network trained to have a small, and acceptable, error rate, the network size need to be expanded such that longer time for each training cycle is required. The neural network has many parameters that need to be set. These parameters will be discussed in the next subsections along with the implementation specifications. The choice of neural networks to be used in this attack is that multilayer feedforward neural networks, which can be used as global approximators as proved in [33].

2.1 Implementation Environment

The software used in implementing the proposed attack was MATLAB R2008a with neural networks toolbox. The single computer used in the implementation had an AMD Athlon X2 processor with 1.9 Gigahertz clock frequency and 4 Gigabytes of memory.

2.2 Design Choices

a. Data representation

Since DES and Triple-DES handle data as blocks of 64 bits, the data representation for both of their cryptanalytic network was the same. The 64-bit blocks were represented as simultaneous 64 inputs to the neural network and the output was also represented as 64 output lines. Obviously, the range of inputs and outputs was [0, 1]. The DES implementation was in Electronic Code Book (ECB) mode, while the triple-DES encryption was in Encryption-Decryption-Encryption (EDE) pattern.

Since the implementation was in MATLAB, the data was represented in the following matrix form:

$$P = \begin{bmatrix} p_{1,1} & p_{1,2} & & p_{1,k} \\ p_{2,1} & p_{2,2} & & p_{2,k} \\ p_{3,1} & p_{3,2} & \cdots & p_{3,k} \\ \vdots & \vdots & & \vdots \\ p_{64,1} & p_{64,2} & & p_{64,k} \end{bmatrix}, C = \begin{bmatrix} c_{1,1} & c_{1,2} & & c_{1,k} \\ c_{2,1} & c_{2,2} & & c_{2,k} \\ c_{3,1} & c_{3,2} & \cdots & c_{3,k} \\ \vdots & \vdots & & \vdots \\ c_{64,1} & c_{64,2} & & c_{64,k} \end{bmatrix}$$

where,

P is the plaintext matrix

C is the ciphertext matrix

$p_{i,j}$ is the i^{th} plaintext bit in the j^{th} plaintext block

$c_{i,j}$ is the i^{th} ciphertext bit in the j^{th} ciphertext block

k is the number of plaintext-ciphertext pairs used in the training

The first column in C is the ciphertext corresponding to the plaintext in the first column in P. The number of rows in the two matrices is chosen based on the number of inputs and outputs of the neural network. All ciphertext in the plaintext-ciphertext pairs are encrypted with the same key. The plaintext blocks and the ciphertext blocks that were encrypted using the same key are called a *dataset*. The average size of datasets used was about 2^{20} plaintext-ciphertext pairs.

The ciphertext in the first column of C is sent to the neural network as the first input, the second column is sent as the second input, and so on. On the other hand, the plaintext in the first column of P is used in the neural network as the reference output for the first input, the second column is used as reference output for the second input and so on.

b. Network size and layout

The neural network size is an essential issue in the success of training. Having a network too large or too small can prevent the network from convergence. No single size is considered suitable for our proposed attack. Few different sizes and arrangements were tried and most of them were successful. A sample of the successfully trained networks will be shown in the results section. In this paper, the network layout will be expressed as

n_1-n_2-n_3-…-n_m

where,

n_i is the number of neurons in the i^{th} layer

m is the number of layers in the neural network

n_m is the number of neurons in the output layer

The input layer is not taken into account in this expression. For the DES and Triple-DES, the topology was chosen to be cascaded-forward network because other feedforward network was not successful in achieving the required cryptanalysis.

c. Training algorithm

The choice was made to use scaled conjugate-gradient as it is a suitable algorithm for large networks. Scaled conjugate-gradient requires usually more training cycles than Levenberg-Marquardt but it requires less memory. On the other hand, when compared to other gradient descent training algorithms, scaled conjugate-gradient is faster to converge.

d. Error function and initial weights

The error function chosen to correct the weights during the training was mean-squared error. The weights and biases are initialized in MATLAB using a function called `initlay`. This function generates randomized matrices for weights and biases.

e. Number of training cycles (epochs)

The maximum numbers of training cycles, or epochs, is set to 10^4.

f. Stopping conditions

Stopping conditions are a group of conditions based on which the training of the neural network stops. When *any* of these conditions is met, the network training stops. The first obvious condition is reaching the final number of training cycles. Another condition is a limit for an acceptable mean-squared error. In this paper we set the error stopping condition to be 10^{-4}. Maximum number of consecutive validation failures is also another stopping condition. In this paper, we set the maximum failures to be 10.

2.3 Methodology

The proposed cryptanalytic attack was implemented on DES using 100 different datasets to prove that this method does not rely on certain properties of the key used in encryption. Each dataset consisted of a certain number of plaintext-ciphertext pairs. For each dataset, all the plaintext in that dataset was encrypted with the same key to obtain the corresponding ciphertext. Each dataset used a key that is different from all other datasets (independent keys). Each dataset was divided into two parts; one part used in the training the neural network, and another part used for validation. The datasets were generated using a Pseudo-random Number (PN) generator and so were the keys. Despite that the data was generated using a PN generator, the data, as well as the key, was checked to assure their independence. For that purpose, a small program was written to remove repeated blocks from both the data and keys. This step was done to insure that there is no bias whatsoever in the plaintext blocks or the keys used in encryption. The following subsections describe the processes used in the cryptanalysis.

Training Process.

1. The training process starts by creating the neural network and selecting its layout and the stopping conditions for the training are then set.

2. The training starts by feeding the neural network with the ciphertext blocks selected from the dataset for training and using the corresponding plaintext blocks as reference output.

3. At the end of training it is decided whether this training trial is a failure or a success. Failure training is identified by reaching one of the stopping conditions with mean-squared-error more than 10^{-2}. On the other hand, successful training was identified by reaching the minimum possible mean-squared error with the minimum possible number of plaintext-ciphertext pairs.

4. If the training fails, the network is re-initialized with random weights and biases and the layout of the network is manually changed and the training is restarted at step 2. Most of the failures are caused by increasing slope of the mean-squared-error due to reaching local minima.

8. If the training was successful, we move to the validation process.

Validation Process.

1. After the training stops, successfully, the same ciphertext matrix used in training is fed into the network to obtain neuro-decrypted-plaintext results; P'. Since the activation function used in all the neurons is non-linear, the output of the neural network is a mostly floating-point numbers. Thus, the output of the neural network is rounded to 0 or 1 to produce data that is comparable to the reference output data.

2. P' is then compared to *P*, bit-by-bit and the percentage of bits in P' having a value different from the corresponding bit in *P* is calculated. This comparison is done using XOR operation. For the sake of simplicity, this error is called *inside-error*. The inside-error can be expressed in the following mathematical form:

$$inside\text{-}error = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} p'(i,j) \oplus p(i,j)}{m \times n} \quad (1)$$

where,

m is the number of bits in a block of training data

n is the number of blocks used in the training

$p'(i,j)$ is the j^{th} bit in the i^{th} block of the neural network output

p(i,j) is the j^{th} bit in the i^{th} block of the plaintext used in training

For both DES and 3DES, m=64

3. Afterwards, more ciphertext is fed into the network; C_{new}. This ciphertext is not part of the ciphertext used in training, but encrypted with the same key. It is important to use data in the validation that is independent from the data used in training. The output is also rounded to 0 or 1 in this step as in step 1.

4. The resulting neuro-decrypted-plaintext, P'_{new} is compared to the original plaintext P_{new} bit-by-bit, and the resulting error percentage is calculated. This error percentage is called *outside-error*. The outside-error can be expressed in the following mathematical formula:

$$outside\text{-}error = \frac{\sum_{i=1}^{m_{new}} \sum_{j=1}^{n_{new}} p'_{new}(i,j) \oplus p_{new}(i,j)}{m_{new} \times n_{new}} \quad (2)$$

where,

m_{new} is the number of bits in a block of validation data

n_{new} is the number of blocks used in the validation

p'_{new} is the jth bit in the ith block of output of the neural network in the validation

$p_{new}(i,j)$ is the jth bit in the ith block of plaintext used in the validation

For both DES and 3DES, $m_{new}=m=64$

5. As mentioned earlier, the ciphertext, C_{new}, and the plaintext P_{new} were chosen from the dataset such that they do not replicate any plaintext-ciphertext pair previously used in training, but the number of blocks in them ranged from 8-10 times the number of blocks used in the training.

Dataset Variation. The same training and validation procedures explained earlier are repeated for each dataset. These repetitions produce a separate neural network for each dataset (i.e. for each key used in encryption). This neural network is capable of retrieving plaintext from, virtually, any ciphertext that was encrypted with the same key used to encrypt the data used in training.

Encryption Algorithm Variation. The exact same processes mentioned previously were implemented on DES-encrypted datasets and Triple-DES-encrypted datasets using 100 different dataset for each algorithm.

3 Results of Implementation

After reaching a successful training trial on each one of the 100 datasets (i.e. producing a trained neural network for each dataset) for both DES and Triple-DES, some calculations were made to get a general tabulation of results. Table 1 shows the general results of implementing the cryptanalytic attack on 100 datasets for DES and 100 data sets for Triple-DES.

Table 1. Results of Implementing Neuro-cryptanalysis on DES and Triple-DES

Calculation	DES	Triple-

		DES
Total Number of trials	833	1093
Number of successful trials	100	100
Number of failure trials	733	993
Average number of plaintext-ciphertext pairs needed for training	2^{11}	2^{12}
Average time of successful training (min.)	51	72
Average time of failure trial (min.)	3	5
Average time until success (min.)	21	52
Average number of trials required to reach success	7.33	9.93
Average inside-error	0.022	0.028
Average outside-error	0.083	0.114

For more detailed results, specific details of the most successful 10 trials in DES and Triple-DES cryptanalysis were provided in Tables 2 and 3.

Table 2. Results of most successful 10 trials on DES

Dataset no.	Net. Layout	Plaintext-ciphertext pairs needed	Inside error	Outside error	Mean-squared error reached	Training time (min.)	Epochs
1	128-256-256-128	2048	0.027973	0.085986	0.013317	39	308
2	128-256-256-128	2048	0.027314	0.099778	0.010491	39	298
3	128-256-256-128	2048	0.029855	0.111682	0.027461	41	334
4	128-256-512-256	2048	0.034162	0.131019	0.015199	42	356
5	128-256-256-128	2048	0.028341	0.099961	0.004963	42	341
6	128-512-256-256	2048	0.037806	0.123977	0.047443	42	351
7	128-256-256-128	2048	0.034402	0.135236	0.035966	42	355
8	128-256-256-128	2048	0.029503	0.085817	0.000685	43	388
9	128-256-512-128	2048	0.041513	0.134885	0.076850	43	379
10	128-256-256-128	2048	0.0267	0.10177	0.075965	43	411

Table 3. Results of most successful 10 trials on Triple-DES

Dataset no.	Net. Layout	Plaintext-ciphertext pairs needed	Inside error (%)	Outside error (%)	Mean-squared error reached	Training time (min.)	Epochs
1	128-512-512-128	4096	0.030316	0.117038	0.043745	62	242
2	128-512-512-128	4096	0.049565	0.210300	0.059068	63	248
3	128-512-512-128	4096	0.047323	0.125995	0.03098	63	239
4	64-128-256-512-1024	4096	0.040790	0.163973	0.037367	63	218
5	128-512-512-128	4096	0.040859	0.174885	0.045597	63	265
6	128-512-512-128	4096	0.053892	0.217162	0.057514	64	233
7	128-256-512-256	4096	0.031928	0.150773	0.023654	64	253
8	128-512-512-128	4096	0.045906	0.194787	0.015597	64	235
9	128-512-512-128	4096	0.033464	0.13131	0.024245	64	229
10	128-256-512-512	4096	0.036299	0.172447	0.035044	64	230

The attack with the least time required for training is considered most successful. All the DES and Triple-DES cryptanalysis neural networks were trained with the scaled-conjugate gradient algorithm, and all of their structures were cascaded-feedforward.

4 Discussions

This attack, as shown in the results tables, has succeeded in getting global deduction for DES with an average less than 2^{11} known-plaintexts. As compared to other attacks, this is the best result reached for a known-plaintext attack on DES. The fact that less than 60 minutes and one computer is enough to break the cipher renders the attack a practically implementable one.

As regarding Triple-DES, the results achieved by this attack are interesting. The need of an average of about 2^{12} known-plaintexts has never been achieved by any other cryptanalytic attack on Triple-DES. Table 4 compares the neuro-cryptanalysis

results to other attacks on DES in terms of required number of plaintext-ciphertext pairs.

Table 4. Number of plaintext-ciphertext pairs required for different attacks on DES.

Attack Type	Neuro-Cryptanalysis	Differential Crypt-analysis	Linear Cryptanaly-sis
Number of pairs required	2^{11}	2^{47}	2^{43}

A downside of this cryptanalytic method is the outer error. It is clear from the tables 2 and 3 that having less plaintext-ciphertext pairs can increase the outer error percentage. Having a time-until-success average of about 21 minutes for DES and 52 minutes for Triple-DES is a noticeable advancement in cryptanalysis of both of these standards.

Since the system is a neural network, its implementation in hardware is fairly possible. This implementation can be done through dedicated neural processors or through other generalized hardware like Field-Programmable Gate Array (FPGA). These implementations can give better performance than the implemented software system. The proposed attack can be implemented on other cryptographic systems as well. Its success relies greatly on the block size of the cipher. Jumping to a cipher of a block size of 128 bits, for example, will cause great enlargement in the neural network size.

5 Conclusions

This paper focuses on presenting a new cryptanalytic attack on DES and Triple-DES based on neural networks. This attack was implemented by training a neural network to retrieve plaintext of the ciphertext fed into it. The attack presented here is a known-plaintext attack that retrieves the plaintext of a given ciphertext without retrieving the key.

The attack was successfully implemented using MATLAB. The successful cryptanalysis of DES required an average of 2^{11} plaintext-ciphertext pairs, while the cryptanalysis of Triple-DES required and average of 2^{12} plaintext-ciphertext pairs. The average time required to train the neural network to perform cryptanalysis of DES was 51 minutes, while this average training time was 72 minutes to achieve Triple-DES cryptanalysis. These results are considered an important achievement regarding the cryptanalysis of DES and Triple-DES. The results obtained for Triple-DES cryptanalysis are particularly important because Triple-DES is still widely used in many commercial applications.

The proposed cryptanalytic attack can be further implemented in hardware to improve its performance. The scope of this attack can also be expanded to other block ciphers and other cryptanalytic techniques.

References:

1. M. Albrecht, and C. Cid, Algebraic Techniques in Differential Cryptanalysis, *Proceedings of the First International Conference on Symbolic Computation and Cryptography*, SCC 2008, Beijing, China, April 2008.
2. L.R. Knudsen, Block Ciphers – a Survey, In B. Preneel and V. Rijmen (editors) *State of the Art in Applied Cryptography*, LNCS 1528, pages 18–48. Springer-Verlag, 1998.
3. H. Fiestel, Cryptography and computer privacy, Scientific American, No. 228, pp. 15-23, 1973.
4. NIST, Data Encryption Standard (DES), FIPS PUB 46, Federal Information Processing Standards Publication 46, U.S. Department of Commerce, Jan 1977.
5. E. Biham, and A. Shamir, Differential Cryptanalysis of DES-like Cryptosystems, *Journal of Cryptology*, Vol. 4, No. 1, pp. 3–72, 1991.
6. NIST, Data Encryption Standard (DES), FIPS PUB 46–3, Federal Information Processing Standards Publication 46–3, Oct 1999.
7. M. Matsui, Linear Cryptanalysis Method for DES Cipher, In T. Helleseth (ed.) Advances in Cryptology - EUROCRYPT'93, LNCS 765, pages 386–397. Springer-Verlag, 1994.
8. E. Biham, and A. Shamir, Differential Cryptanalysis of the Full 16-Round DES, In E.F. Brickell (ed.), *Advances in Cryptology, Crypto'92*, LNCS 740, pp. 487–496. Springer-Verlag, 1993.
9. E. Biham, A. Biryukov, and A. Shamir, *Cryptanalysis of Skipjack Reduced to 31 Rounds using Impossible Differentials*, Tech Report CS0947 revised, Technion, CS Dept., 1998.
10. S.K. Langford, *Differential-Linear Cryptanalysis and Threshold Signatures*, PhD thesis, Stanford University, USA, 1995.
11. E. Biham, New Types of Cryptanalytic Attacks using Related Keys, In T. Helleseth (ed.) Advances in Cryptology EUROCRYPT'93, LNCS 765, pp. 398–409. Springer-Verlag, 1994.
12. L.R. Knudsen, Cryptanalysis of LOKI, In H. Imai, R.L. Rivest, and T. Matsumoto (Ed.s) *Advances in Cryptology - Asiacrypt'91*, LNCS 739, pages 22–35. Springer-Verlag, 1993.
13. T. Shimoyama, and T. Kaneko, Quadratic Relation of S-box and its Application to the Linear Attack of Full Round DES, In H. Krawczyk (Ed.) *Advances in Cryptology - Crypto'98*, LNCS 1462, pages 200–211. Springer-Verlag, 1998.
14. R. Merkle, M. Hellman, On the Security of Multiple Encryption, *Communications of the ACM*, Vol. 24, No 7, pp. 465–467, July 1981.
15. P. van Oorschot, and M. J. Wiener, A known-plaintext attack on two-key triple encryption, in I. Damgard (Ed.) *Advances in Cryptology - EUROCRYPT'90*, LNCS 473, pp 318–325, Springer-Verlag, 1991.
16. W. Diffie, and M. E. Hellman, Exhaustive Cryptanalysis of the NBS Data Encryption Standard, *Computer*, Vol. 10, No. 6, pp.74–84, June 1977.
17. S. Lucks, Attacking Triple Encryption, in S. Vaudenay (Ed.) *Fast Software Encryption - FSE'98*, LNCS 1372, pp. 239-253, Springer-Verlag, 1998.
18. J. Kelsey, B. Schneier, D. Wagner, Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES, in N. Koblitz (Ed.) *Advances in Cryptology - CRYPTO '96*, LNCS 1109, pp.237-251, 1996.
19. P. van Oorschot, and M.J. Wiener, Parallel Collision Search with Cryptanalytic Applications, *Journal of Cryptology*, Vol.12, No.1, pp.1–28, 1999.
20. J. Choi, J. Kim, J. Sung, S. Lee and J. Lim, Related-Key and Meet-in-the-Middle Attacks on Triple-DES and DES-EXE, in O.Gervasi et al. (Eds.) *Proceeding of ICCSA 2005*, LNCS 3481, pp. 567-576, 2005,Springer-Verlag, Germany, 2005.

21. M. Clark, and D. Blank, A Neural-Network Based Cryptographic System, *Procceedings of the 9th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS'98)*, pp. 91-94 Dayton, USA, 1998.
22. T. Godhavari, N.R. Alamelu, R. Soundararajan, Cryptography Using Neural Network, *Proceedings of 2005 Annual IEEE INDICON*, pp. 258-261, 2005.
23. Kanter, W. Kinzel, E. Kanter, Secure exchange of information by synchronization of neural networks, *Europhysics Letters*, Vol. 57, No. 1, page 141, 2002.
24. A.Klimov, A.Mityagin, A. Shamir, Analysis of Neural Cryptography, in Y.Zheng (Ed.) *Advance in Cryptology - ASIACRYPT 2002*, LNCS 2501, pp. 288-298, Springer, Berlin, Germany, 2002.
25. L. Li, L. Lin, M. Hwang, A remote password authentication scheme for multiserver architecture using neural networks, *IEEE Transactions on Neural Networks*, Vol. 12, No. 6, pp.1498-1504, Nov. 2001.
26. S. Dourlens, *Neuro-Cryptography*, MSc Thesis, Dept. of Microcomputers and Microelectronics, University of Paris, France, 1995.
27. Ruttor, W. Kinzel, R. Naeh, I. Kanter, Genetic attack on neural cryptography, *Physics Review E*, Vol. 73, No. 3, pp. 121-129, March, 2006.
28. S. Li, *Analyses and New Designs of Digital Chaotic Ciphers*, PhD thesis, School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an, China, 2003.
29. C. Li, S. Li, D. Zhang, G. Chen, Chosen-Plaintext Cryptanalysis of a Clipped-Neural-Network-Based Chaotic Cipher, *Proceedings of International Symposium on Neural Networks*, China, 2005.
30. K. Rao, M. Krishna, D. Babu, Cryptanalysis of a Feistel Type Block Cipher by Feed Forward Neural Network Using Right Sigmoidal Signal, *International Journal of Soft Computing*, Vol.4, No.3, pp. 131-135, 2009.
31. B. Chandra, and P. Paul Varghese, Applications of Cascade Correlation Neural Networks for Cipher System Identification, World Academy of Science, Engineering and Technology, Issue 26, 2007.
32. M. M. Alani, Neurocryptanalysis of DES, Ptoceedings of World Congress on Internet Security 2012, University of Guelph, Guelph, Canada, June 2012.
33. K. Hornik, M. Stinchcombe, H. White, Multilayer Feedforward Neural Networks are Universal Approximators, *Neural Networks*, Vol. 2, pp. 359-366, 1989.