ARTICLE IN PRESS

Available online at www.sciencedirect.com

ScienceDirect

Chaos, Solitons and Fractals xxx (2007) xxx–xxx

CHAOS SOLITONS & FRACTALS

www.elsevier.com/locate/chaos

Cryptanalysis of a cryptographic scheme based on delayed chaotic neural networks

Jiyun Yang [a,*], Xiaofeng Liao [a,b], Wenwu Yu [c], Kwok-wo Wong [d], Jun Wei [e]

[a] *Department of Computer Science and Engineering, Chongqing University, Chongqing 400044, PR China*
[b] *The Key laboratory of Optoelectric Technology and Systems, Ministry of Education, PR China*
[c] *Department of Mathematics, Southeast University, Nanjing 210096, PR China*
[d] *Department of Computer Engineering and Information Technology, City University of Hong Kong, Hong Kong*
[e] *Zhunyi Medical College, Zhunyi 563000, Guizhou, PR China*

Accepted 13 August 2007

Abstract

Recently, Yu et al. presented a new cryptographic scheme based on delayed chaotic neural networks. In this letter, a fundamental flaw in Yu's scheme is described. By means of chosen plaintext attack, the secret keystream used can easily be obtained.

1. Introduction

Chaotic systems possess many interesting properties such as ergodicity, mixing and sensitivity to initial conditions which match with the requirements for a good cryptosystem. Recently, there are an increasing number of researchers working in this field, resulting in a variety of designs of cryptosystems based on chaotic systems [1–3,5].

Recently, Yu et al. designed a cryptosystem based on delayed chaotic neural networks [6]. This cryptosystem makes use of the chaotic trajectories of two neurons to generate basic binary sequences for encrypting plaintext according to some rules. Yu et al. claimed that it is difficult to synchronize the unknown chaotic neural networks through classical attacks since neural networks usually possess complicated parameters [6]. However, a detailed analysis on the encryption algorithm shows that the cipher behaves as a stream cipher indeed [9, p. 20] although it looks like a block cipher. Moreover, every new encryption process has the same basic binary sequences. This weakness leads to the re-generation of the keystream under chosen plaintext attacks. By the fact that knowing the keystream generated by certain neural networks is equivalent to knowing the parameters of the neural networks, the cipher is broken.

In this letter, we will first give a brief introduction to Yu et al.'s cryptographic scheme. Then the flaw of this scheme will be analyzed in detail. The method on how to obtain the keystream under chosen plaintext attacks will also be described. Finally, a conclusion of our findings will be drawn.

* Corresponding author.
E-mail address: yangjy@cqu.edu.cn (J. Yang).

0960-0779/$ - see front matter
doi:10.1016/j.chaos.2007.08.029

2. A brief introduction to Yu's cryptography

Yu et al.'s cryptosystem is governed by the following Hopfield neural networks [6]:

$$\begin{pmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \end{pmatrix} = -A\begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + W\begin{pmatrix} \tanh(x_1(t)) \\ \tanh(x_2(t)) \end{pmatrix} + B\begin{pmatrix} \tanh(x_1(t-\tau(t))) \\ \tanh(x_2(t-\tau(t))) \end{pmatrix} \tag{1}$$

where $\tau(t) = 1 + 0.1\sin(t)$, the initial condition of (1) is given by $x_i(t) = \phi_i(t)$ when $-r \leqslant t \leqslant 0$, where $r = \max_{t\in R}\{\tau(t)\}$, $\phi(t) = (0.4, 0.6)^{\mathrm{T}}$.

The set of delayed differential equations is solved by the fourth-order Runge–Kutta method with time step size $h = 0.01$. Suppose that $x_1(t)$ and $x_2(t)$ are the trajectories of delayed neural networks (1). The ith iterations of the chaotic neural networks are $x_{1i} = x_1(ih)$, $x_{2i} = x_2(ih)$.

In Yu et al.'s cryptosystem, an approach proposed in [10] was adopted to generate a sequence of independent and identical (i.i.d.) binary random variables from a class of ergodic chaotic maps. For any x defined in the interval $I = [d, e]$, we can express the value of $(x-d)/(e-d) \in [0, 1]$ in the following binary representation:

$$\frac{x-d}{e-d} = 0.\quad b_1(x)b_2(x)\ldots b_i(x)\ldots,\qquad x \in [d,e],\quad b_i(x) \in \{0,1\}. \tag{2}$$

The ith bit $b_i(x)$ can be expressed as

$$b_i(x) = \sum_{r=1}^{2^i-1}(-1)^{r-1}\Theta_{(e-d)(r/2^i)+d}(x) \tag{3}$$

where $\Theta_t(x)$ is a threshold function defined by

$$\Theta_t(x) = \begin{cases} 0, & x < t \\ 1, & x \geqslant t \end{cases} \tag{4}$$

By Eq. (3), a binary sequence $B_i^k = \{b_i(x_k)\}_{k=0}^{\infty}$ is obtained, where x_k is the kth iteration of the chaotic neural networks (1).

After the basic binary sequence is generated by Eqs. (1)–(4), it can be used for encryption according to the following procedures:

Step 1. Get the start point x_0 from the last N_0 transient iterations, $x_0 = x_1(N_0h)$. In this scheme, N_0 is chosen as 1000.

Step 2. Divide the message p into subsequences P_j of length l bytes. In this scheme l is chosen as 4. $P_j = p_{lj} + p_{lj+1} + p_{lj+2} + p_{lj+3}$, where + denotes concatenation.

Step 3. Iterate neural networks (1) for 38 times to generate two data sequences: $x_1 = x_{10}x_{11}\ldots x_{137}$ and $x_2 = x_{20}x_{21}\ldots x_{237}$. Choose one of these data sequences to generate the binary sequence $A_j = B_i^1B_i^2\ldots B_i^{32}$, $D_j = B_i^{33}B_i^{34}\ldots B_i^{37}$, $S_j = B_i^{38}$ based on Eq. (3), where $i = 4$. The choice is governed by the following rule: If the first four bytes of the message sequence are being encrypted, choose x_1 sequence. Otherwise choose the data sequence according to the previous S_j. If $S_j = 0$, choose the x_1 sequence. Otherwise, use the x_2 sequence.

Step 4. Left cyclic shift the message block P_j for D_j bits and right cyclic shift block A_j for D_j bits to generate P_j' and A_j', respectively.

Step 5. Use P_j' and A_j' to generate C_j according to the following equation:

$$C_j = P_j' \oplus A_j' \tag{5}$$

where $\oplus$ is XOR operation.

Step 6. If all plaintext blocks have already been encrypted, the encryption process is completed. Otherwise, let $x_0 = x_{S_j+1}((38 + D_j)h)$, and go to step 2.

The decryption process is the same as the encryption one except that the shifted message block is obtained by

$$P_j' = C_j \oplus A_j' \tag{6}$$

For more details, we highly suggest a thorough reading of [6].

Please cite this article in press as: Yang J et al., Cryptanalysis of a cryptographic scheme based ..., Chaos, Solitons & Fractals (2007), doi:10.1016/j.chaos.2007.08.029

3. Analysis on Yu et al.'s cryptosystem

However, Yu's scheme is found to have a fundamental flaw. As long as the key is fixed, the keystream A'_j used in Eq. (5) is independent of the plaintext. Then every new encryption process will be based on the same keystream. When this algorithm is used to encrypt identical plaintexts at the same encryption position, identical ciphertexts are generated. This situation will occur frequently, especially when encrypting files are of the same type. This is because those files usually have the same header.

In step 3 of Yu et al.'s encryption algorithm, i is usually set to a relatively small value, i.e., a relatively heavy weight bit, A_j, D_j and S_j vary little in the encryption process.

In order to illustrate this flaw, herein we give a part of the keystream according to Yu et al.'s encryption algorithm. In [6] the parameters of neural networks (1) are chosen as $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $W = \begin{pmatrix} 2.0 & -0.1 \\ -5.0 & 3.0 \end{pmatrix}$, $B = \begin{pmatrix} -1.5 & -0.1 \\ -0.2 & -2.5 \end{pmatrix}$, $h = 0.01$, $\phi(t) = (0.4, 0.6)^{\mathrm{T}}$ in the time interval $[-1.1, 0]$.

Table 1 shows the first 10 keystreams obtained using Yu et al.'s encryption algorithm.

In every new encryption process, the same keystream is employed to encrypt the plaintext. To demonstrate this situation, the results of encrypting two different plaintext sequences are shown in Tables 2 and 3. The two plaintext sequences, P1 and P2, are arbitrarily generated as '512F7D86109A32C436AB95C28901A023' and '60A398C2016540238AC0365236DC01B2', respectively, expressed in hexadecimal format.

Table 1
A part of keystream

Encryption position	Transient N_0	Trajectory	Start point x_0	A_j (Hex)	D_j (Hex)	A'_j (Hex)	S_j (B)
1	1000	x_1	−0.368	FFFFFFFF	F	FFFFFFFF	1
2	53	x_2	2.245	FFFFFFFF	F	FFFFFFFF	1
3	53	x_2	1.6205	00000000	0	00000000	0
4	38	x_1	−0.2045	0FFFFFFF	F	FFFE1FFF	1
5	53	x_2	1.9577	FFFFFFFF	F	FFFFFFFF	1
6	53	x_2	2.8385	00000000	0	00000000	0
7	38	x_1	−0.64754	0007FFFF	F	FFFE000F	1
8	53	x_2	3.1357	C0000000	0	C0000000	0
9	38	x_1	−0.33235	00000000	0	00000000	0
10	38	x_1	−0.90302	0007FFFF	F	FFFE000F	1
...	...	...	...	...	...	...	...

Table 2
Encryption process for P_1

Encryption position	Plaintext P_j (Hex)	D_j (Hex)	P'_j (Hex)	A'_j (Hex)	C_j (Hex)
1	512F7D86	F	BEC32879	FFFFFFFF	413CD768
2	109A32C4	F	193A084D	FFFFFFFF	E6C5F7B2
3	36AB95C2	0	36AB95C2	00000000	36AB95C2
4	8901A023	F	D011C480	FFFE1FFF	2FEFDB7F

Table 3
Encryption process P_2

Encryption position	Plaintext P_j (Hex)	D_j (Hex)	P'_j (Hex)	A'_j (Hex)	C_j (Hex)
1	60A398C2	F	CC613051	FFFFFFFF	339ECFAE
2	01654023	F	A0118DB2	FFFFFFFF	5FEE724D
3	8AC03652	0	8AC03652	00000000	8AC03652
4	36DC01B2	F	00D91B6E	FFFE1FFF	FF270491

Please cite this article in press as: Yang J et al., Cryptanalysis of a cryptographic scheme based ..., Chaos, Solitons & Fractals (2007), doi:10.1016/j.chaos.2007.08.029

4. Chosen plaintext attack

There are four different levels of attacks presented in [9, p. 25] to test the security of various encryption algorithms. Ordered by difficulty, they are, respectively, ciphertext-only attack, known plaintext attack, chosen plaintext attack, and chosen ciphertext attack. If an algorithm is resistant to all known attacks under the assumption that the cryptanalyst has the details of the algorithm [9, p. 25], it is proven to be secure.

The chosen plaintext attack on the cryptosystem proposed in [6] is straightforward. Suppose that two pairs of plaintext and the corresponding ciphertext with the same encryption position and desired length are obtained. Let P_{j1} and C_{j1}, respectively, denote the plaintext and the ciphertext of the first pair while P_{j2} and C_{j2} denote another pair. Since P_{j1} and P_{j2} are located at the same position j in the encryption process, they will left cyclic shift by the number of bits D_j and are processed with the same keystream A'_j.

From step 4 of the encryption algorithm, we know

$$P'_{j1} = P_{j1} \ll D_j \tag{7}$$

$$P'_{j2} = P_{j2} \ll D_j \tag{8}$$

where $\ll$ denotes the left cyclic shift operation.

From step 5, we know

$$\begin{aligned} C_{j1} &= P'_{j1} \oplus A'_j \\ C_{j2} &= P'_{j2} \oplus A'_j \\ C_{j1} \oplus C_{j2} &= P'_{j1} \oplus A'_j \oplus P'_{j2} \oplus A'_j \\ C_{j1} \oplus C_{j2} &= P'_{j1} \oplus P'_{j2} \end{aligned} \tag{9}$$

From Eqs. (7) and (8), we obtain

$$C_{j1} \oplus C_{j2} = (P_{j1} \oplus P_{j2}) \ll D_j \tag{10}$$

Let $X_j = C_{j1} \oplus C_{j2}$, $Z_j = P_{j1} \oplus P_{j2}$.

Since X_j and Y_j are known, in order to get D_j, we only need to left cyclic shift Z_j until $X_j = Z_j$. At that time, the number of shifts is D_j. Of course, we hope there exists a unique solution for Eq. (10). However, if Z_j derives the type of 'aaa...a', then there exist more than one solutions for Eq. (10) because $Z_j \ll n = Z_j \ll n + k * L$ with 'a' having L bits. So we cannot chose plaintext pairs which let Z_j have the type of 'aaa...a'.

When D_j is obtained, we can obtain P'_{j1} from Eq. (7).

From Eq. (9)

$$A'_j = P'_{j1} \oplus C_{j1} \tag{11}$$

Following this computationally inexpensive method, we can obtain the desirable keystream A'_j. It is important to note that knowing the keystream A'_j generated by a certain key is equivalent to knowing the secret key [4,7,8].

To demonstrate the security loophole caused by this flaw, we choose two plaintexts from Tables 2 and 3 at the fourth block, with P_{41} and P_{42} denoting them, respectively. The processes of chosen plaintext attack to the cryptosystem are listed step by step as follows:

Step 1. From Tables 2 and 3, we get $P_{41} = $ 8901A023H, $P_{42} = $ 36DC01B2H, $P'_{41} = $ D011C480H, $P'_{42} = $ 00D91B6EH, $C_{41} = $ 2FEFDB7F H and $C_{42} = $ FF270491H.

Step 2. Suppose we only know P_{41}, P_{42}, C_{41}, and C_{42}. According to Eqs. (7), (8) and (10), we compute D_j, A'_j and A_j to validate our result.
 (i) $C_{41} \oplus C_{42} = $ D0C8DFEEH, and denote it as X_4.
 (ii) $P_{41} \oplus P_{42} = $ BFDDA191H, and denote it as Z_4.
 (iii) By left cyclic shifting Z_4 until $X_4 = Z_4$, we can obtain the number of shifts $D_4 = $ FH which is the same as the value listed in Tables 2 and 3.
 (iv) According to Step 4 of Yu et al.'s algorithm, we can obtain $P'_{41} = $ D011C480H and $P'_{42} = $ 00D91B6EH.
 (v) According to Eq. (11), we can obtain $A'_4 = $ FFFE1FFF H which is the same as the value listed in Tables 2 and 3.
 (vi) According to Step 4 of Yu et al.'s algorithm, we get $A_4 = $ 0FFFFFFFH. It is also the same as the value listed in Table 1.

From the above demonstration, we can easily obtain the keystream using only two pairs of plaintext and ciphertext.

5. Conclusion

In this letter, Yu et al.'s cryptosystem as proposed in [6] is analyzed in detail. It is difficult to obtain the key of Yu et al.'s cryptosystem through classical attacks because of large key space. However, as the same keystream is used in every encryption process, it can be easily obtained by the chosen plaintext attack using two pairs of plaintext and ciphertexts only. This makes this cryptographic scheme insecure.

Acknowledgements

The work described in this paper was supported by the National Natural Science Foundation of China (No. 60574024), New Century Excellent Talents in University, and Natural Science Foundation of Chongqing (No. 8414).

References

[1] Baptista MS. Cryptography with chaos. Phys Lett A 1998;240:50–4.
[2] Wei Jun, Liao Xiaofeng, et al. A new chaotic cryptosystem. Chaos, Solitons & Fractals 2006;30:1143–52.
[3] Gao Haojiang, Zhang Yisheng, et al. A new chaotic algorithm for image encryption. Chaos, Solitons & Fractals 2006;29:393–9.
[4] Zhang Linhua, Liao Xiaofeng, et al. An image encryption approach based on chaotic maps. Chaos, Solitons & Fractals 2005;24:759–65.
[5] Huang Fangjun, Guan Zhi-Hong. A modified method of a class of recently presented cryptosystems. Chaos, Solitons & Fractals 2005;23:1893–9.
[6] Yu W, Cao J. Cryptography based on delayed neural networks. Phys Lett A 2006;356:333–8.
[7] Lei Min, Meng Guang, et al. Security analysis of chaotic communication systems based on Volterra–Wiener–Korenberg model. Chaos, Solitons & Fractals 2006;28:264–70.
[8] Álvarez G, Montoya F, et al. Cryptanalyzing an improved security modulated chaotic encryption scheme using ciphertext absolute value. Chaos, Solitons & Fractals 2005;23:1749–56.
[9] Stinson DR. Cryptography: theory and practice. Boca Raton, FL: CRC Press; 1995.
[10] Wu Xiaogang, Hu Hanping, et al. Analyzing and improving a chaotic encryption method. Chaos, Solitons & Fractals 2004;22:367–73.