INTERNATIONAL JOURNAL OF CIRCUITS, SYSTEMS AND SIGNAL PROCESSING
DOI: 10.46300/9106.2022.16.111 Volume 16, 2022

A Formal Verification Based on Yu-Cao Delayed Chaotic Neural Network

Chi Huang[1], Chenglian Liu[1], Yueyang Cai[2,a], Sonia C-I Chen[2,b], Xiaofei Ji[3,c]
[1]Department of Science and Engineering, Shiyuan College of Nanning Normal University, Nanning 530226, China
[2]School of Economics, Qingdao University, Qingdao 266061, China
[3]School of Computer Science and Technology, Qingdao University, Qingdao 266071, China

Received: March 20, 2021. Revised: February 19, 2022. Accepted: March 8, 2022. Published: March 28, 2022.

***Abstract*—Yu and Cao proposed "Cryptography based on delayed chaotic neural networks" in 2006. However, in 2009, Yang et al. pointed out the Yu-Cao scheme can not against chosen plaintext attack. Liu et al. studies exclusive-or logical operation very well, and provided Boolean algebra proofs in 2012. Ye et al. used Liu et al.'s method to reinterpret and analyze Yu-Cao scheme in 2018. In this paper the authors would like to give a formal verification by Galois field expression on the exclusive-or operation problem again. As this result, it makes more effective insecure to Yu-Cao algorithm.**

***Index Terms*—Neural Network, Chaotic Cryptosystem, Boolean Algebra, Exclusive-OR Operation**

1. Introduction

Chaos theory has brought signif cant inf uences on cryptography and computer science in the past decades. The variety of chaos theory diffuses various types of encryptions, secret key or symmetric key [1]. Yu and Cao [2] proposed a novel approach of encryption employed chaotic Hopf eld neural networks with time-varying delay. This concept describes generating binary sequences for encrypting plaintext according to the rules. Yang et al. [3] discovered a fundamental f aw in the Yu-Cao scheme and gave a method to fetch the keystream by choosing plaintext attacks in 2009. Liu et al. [4] had a comprehensive study on the exclusive-or (XOR) topic that they showed the singular problems in which two variants do bitwise exclusive-or operation. Later, Ye et al. [5] applied Liu et al.'s method to analyze the Yu-Cao scheme in 2018. Although there are some substantial contributions have been made in the combination of XOR and chaotic theory [1], [6]–[8] or neural network f elds [5], and connected applications [9]–[15], this study focuses on how delayed chaotic neural networks can be applied to formal verif cation. The general comparisons of related research can be seen in Table 1. This research is mapped as followed: Section 2 reviews the Yu-Cao scheme. Followed by it, section 3 investigates Yang et al.'s method. Next, the authors' viewpoint of inferences is introduced and discussed. The conclusion is drawn in the f nal section.

TABLE 1
Related Literatures

year	Chaotic neural-network	Chaotic image	Chaotic map	Others
2006			Xiang et al. [16]	
2006	Yu and Cao [2]			
2007			Wang et al. [17]	
2009	Yang et al. [3]			
2009			Wang and Yu [18]	
2013		Li et al. [19]		
2018				Ye et al. [5]
2019				Garcia [9]
2020				Chen et al. [12]
2020				Kanaan et al. [10]
2020				Yang et al. [11]
2020				Al-Mawsawi et al. [13]
2021				Sultana et al. [14]
2021				Voloşencu [15]

2. Review of Yu-Cao Scheme

The Yu-Cao cryptosystem is governed by the following Hopf eld neural networks [2]:

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

where $\tau(t) = 1 + 0.1\sin(t)$, the initial condition of (2) is given by $x_i(t) = \phi_i(t)$ when $-r \leq t \leq 0$, where $r = \max_{t\in R}\{\tau(t)\}, \phi(t) = 0.4, 0.6)^T$. 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 (2). The i-th iterations of the chaotic neural networks are $x_{1i} = x_1(ih), x_{2i}(ih)$. In the Yu-Cao cryptosystem, an approach proposed in [20] 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:

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

The i-th 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 $\Theta_i(x)$ is a threshold function defined by

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

By Equation (3), a binary sequence $B_i^k = \{b_i(x_k)\}_{k=0}^{\infty}$ is obtained, where x_k is the k-th iteration of the chaotic neural networks by Equation (2). After the basic binary sequence is generated by Equation (2) to (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 Equation (2) for 38 times to generate two data sequences: $x_1 = x_{10}x_{11}\cdots x_{137}$ and $x_2 = x_{20}x_{21}\cdots x_{237}$. Choose one of these data sequences to generate the binary sequence $A_j = B_i^1 B_i^2 \cdots B_i^{32}$, $D_j = B_i^{33} B_i^{34} \cdots B_i^{37}$, $S_j = B_i^{38}$ based on Equation (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. P'_j and A'_j to generate C_j according to the following equation:

$$C_j = P'_j \oplus A'_j. \tag{5}$$

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 [2].

3. Security analysis

3.1. Yang et al.'s method

The Yu-Cao scheme is found to have a fundamental flaw by Yang et al. [3]. As long as the key is fixed, the keystream A'_j on the Equation (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 the Yu-Cao encryption algorithm, the 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. From Step 4 of the encryption algorithm, we know

$$\begin{aligned} P'_{j1} &= P_{j1} \ll D_j && (7)\\ P'_{j2} &= P_{j2} \ll D_j && (8)\\ C_{j1} \oplus C_{j2} &= P'_{j1} \oplus A'_j \oplus P'_{j2} \oplus A'_j = P'_{j1} \oplus P'_{j2} && (9)\\ C_{j1} \oplus C_{j2} &= (P_{j1} \oplus P_{j2}) \ll D_j && (10)\\ A'_j &= P'_{j1} \oplus C_{j1} && (11) \end{aligned}$$

The processes of chosen plaintext attack to the cryptosystem are listed step by step as follows:

Step B1. We get $P_{41} = 8907A023h$, $P_{42} = 36DC01B2h$, $P'_{41} = D011C480h$, $P'_{42} = 00D91B6Eh$, $C_{41} = FFEFDB7Fh$ and $C_{42} = FF270491h$.

Step B2. Assume we only know P_{41}, P_{42}, C_{41} and C_{42}, and compute D_j, A'_j and A_j as follow:

1) denote $X_4 = C_{41} \oplus C_{42} = D0C8DFEEh$, and $Z_4 = P_{41} \oplus P_{42} = BFDDA191h$.

2) By left cyclic shifting Z_4 until $Z_4 = X_4$, we can obtain the number of shifts $D_4 = Fh$.
3) According to Step 4 of Yu-Cao algorithm, we can obtain $P'_{41} = D011C480h$ and $P'_{42} = 00D91B6Eh$.
4) According from $A'_j = P'_{j1} \oplus C_{j1}$, we can obtain $A'_4 = FFFE1FFFh$.
5) According to Step 4 of Yu-Cao algorithm, we get $A_4 = 0FFFFFFFh$.

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

3.2. Our methodology

In this subsection, we will point out a leak for Yang et al.'s method and our attack. Even if Yang et al. scheme used two pairs of plaintext and ciphertexts to obtain the keystream of Yu-Cao cryptosystem and attack it. Yang et al. used bitwise exclusive-or (XOR) operation to compute X_4, Z_4 and we can denote $X_4 = C_{41} \oplus C_{42}$, $Z_4 = P_{41} \oplus P_{42}$. In fact, the XOR operation is a common component in the design of digital logical. It is used on adder, cryptosystem or other applications. We derived the Lemma 1 to Lemma 4 [4]:

1. The Boolean Algebra Expression

Table 2 describes the XOR truth table. The XOR operation can be expressed as

$$A \oplus B = (\neg A \wedge B) \vee (A \wedge \neg B), \tag{12}$$

or express to

$$B \oplus A = B\overline{A} + \overline{B}A. \tag{13}$$

It is a very common component in digital circuit or logic, and often used to many f elds such as adder, cryptosystem, image process and so on.

TABLE 2
THE XOR TRUTH TABLE

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Lemma 1. *Let $\oplus$ be an operation on the set X. It is called commutative if $A \oplus B$=$B \oplus A$ for all $A, B \in X$.*

Proof. To prove $A \oplus B = \overline{A}B + A\overline{B}$ where $B \oplus A = B\overline{A} + \overline{B}A$, therefore $\overline{A}B + A\overline{B} = B\overline{A} + \overline{B}A$.
We obtain $A \oplus B$=$B \oplus A$.
Thus, the XOR matches commutative law. □

Lemma 2. *Let $\oplus$ be an operation in the set X. It is called associative if $(A \oplus B) \oplus C = A \oplus (B \oplus C)$ for all $A, B \in X$.*

Proof. $= (A \oplus B) \oplus C = (\overline{A}B + A\overline{B}) \oplus C$.
$= (\overline{A}B + A\overline{B}) \oplus C$.
$= \overline{(A\overline{B} + \overline{A}B)}C + (A\overline{B} + \overline{A}B)\overline{C}$.
$= \overline{(A\overline{B} + \overline{A}B)}C + (A\overline{B} + \overline{A}B)\overline{C}$.
$= \overline{(\overline{A}B) \cdot (A\overline{B})}C + \overline{A}B\overline{C} + A\overline{B} \cdot \overline{C}$
$= (A + \overline{B})(\overline{A} + B)C + \overline{A}B\overline{C} + A\overline{B} \cdot \overline{C}$
$= A\overline{A}C + ABC + \overline{B} \cdot \overline{A}C + \overline{B}BC + \overline{A}B\overline{C} + A\overline{B} \cdot \overline{C}$
$A\overline{A} = 0$ and $B\overline{B} = 0$
$= ABC + \overline{B} \cdot \overline{A}C + \overline{A}B\overline{C} + A\overline{B} \cdot \overline{C}$
Computing $A \oplus (B \oplus C)$
$= A \oplus (B \oplus C) = A \oplus (\overline{B}C + B\overline{C})$
$= \overline{A}(\overline{B}C + B\overline{C}) + A\overline{(\overline{B}C + B\overline{C})}$
$= ABC + ABC + A\overline{(\overline{B}C)} \cdot \overline{(B\overline{C})}$
$= \overline{A} \cdot \overline{B}C + \overline{A}B\overline{C} + A(B + \overline{C})(\overline{B} + C)$
$= \overline{A} \cdot \overline{B}C + \overline{A}B\overline{C} + AB\overline{B} + A\overline{B} \cdot \overline{C} + ACB + A\overline{C}C$
$= \overline{A} \cdot \overline{B}C + \overline{A}B\overline{C} + ABC + A\overline{C} \cdot \overline{B}$.
Since $ABC + \overline{B} \cdot \overline{A}C + \overline{A}B\overline{C} + A\overline{B} \cdot \overline{C} = \overline{A} \cdot \overline{B}C + \overline{A}B\overline{C} + ABC + A\overline{C} \cdot \overline{B}$.
Thus, $(A \oplus B) \oplus C = A \oplus (B \oplus C)$.
Here, the XOR matches associative law. □

Lemma 3. *Let $A = B$, $A \oplus B = \overbrace{0000 \ldots 0000}^{bits}$.*

Proof. As known from Table 2, we get $A \oplus A = 0$, therefore $A \oplus B = \overbrace{0000 \ldots 0000}^{bits}$. □

Lemma 4. *If A and B are both odd numbers, where $(A) \oplus (-A) = \overbrace{1111 \ldots 1110}^{bits}$, $(B) \oplus (-B) = \overbrace{1111 \ldots 1110}^{bits}$, then $(A \oplus B) = (-A \oplus -B)$.*

Proof. According to Lemma 3, if $A = B$, then $(A \oplus B) \oplus (-A \oplus -B) = \overbrace{0000 \ldots 0000}^{bits}$. From Lemma 1 commutative law and Lemma 2 associative law, we rewrite this equation $(A \oplus B) \oplus (-A \oplus -B) = (A \oplus -A) \oplus (B \oplus -B)$. According to Lemma 4, $A \oplus -A = B \oplus -B$.
From Lemma 3, $(A \oplus B) \oplus (-A \oplus -B) = \overbrace{0000 \ldots 0000}^{bits}$.
Therefore, $(A \oplus B) = (-A \oplus -B)$. □

2. The Galois Field Expression

From above, the proof of Lemma 1 to Lemma 4 is described by Boolean algebra. In this paragraph, the authors borrowed binary concept which it re-express to Galios field form, and one theorem integrated four lemmas.

Theorem 1. *If* $2^m \| A$, $2^m \| B$, *then* $(A \oplus B) \equiv (-A \oplus -B) \bmod 2^n$ *, since* $m < n \in N$, $A \in N$ *and* $B \in N$.

Proof. As know $2^m \| A$, $2^m \| B$, we get $A = a_i \sum_{i=m+1}^{n-1} 2^i + 2^m$, $B = b_i \sum_{i=m+1}^{n-1} 2^i + 2^m$ where $a_i \in GF(2)$, $b_i \in GF(2)$. Suppose $-A \equiv (2^2 - A) \bmod 2^n$, namely

$$\begin{aligned} -A &\equiv (2^n - A) \bmod 2^n \\ &\equiv (\sum_{i=0}^{n-1} 2^i + 1 - A) \bmod 2^n \\ &\equiv (\sum_{i=0}^{n-1} 2^i - a_i \sum_{i=m+1}^{n-1} 2^i - 2^m + 1) \bmod 2^n \\ &\equiv (\bar{a}_i \sum_{i=m+1}^{n-1} 2^i + \sum_{i=0}^{m} 2^i - 2^m + 1) \bmod 2^n \\ &\equiv (\bar{a}_i \sum_{i=m+1}^{n-1} 2^i + 2^m) \bmod 2^n. \end{aligned} \tag{14}$$

Similarly, $-B \equiv (\bar{b}_i \sum_{i=m+1}^{n-1} 2^i + 2^m) \bmod 2^n$. We express $(A \oplus B) \equiv \left((a_i \oplus b_i) \sum_{i=m+1}^{n-1} 2^i\right) \bmod 2^n$, and rewrite as

$$\begin{aligned} &(-A \oplus -B) \\ &\equiv \left((\bar{a}_i \sum_{i=m+1}^{n-1} 2^i + 2^m) \oplus (\bar{b}_i \sum_{i=m+1}^{n-1} 2^i + 2^m) \right) \bmod 2^n \\ &\equiv \left((\bar{a}_i \sum_{i=m+1}^{n-1} 2^i) \oplus (\bar{b}_i \sum_{i=m+1}^{n-1} 2^i) + (2^m \oplus 2^m) \right) \bmod 2^n \\ &\equiv \left((\bar{a}_i \oplus \bar{b}_i) \sum_{i=m+1}^{n-1} 2^i \right) \bmod 2^n \\ &\equiv \left((a_i \oplus b_i) \sum_{i=m+1}^{n-1} 2^i \right) \bmod 2^n \\ &\equiv (A \oplus B) \bmod 2^n. \end{aligned} \tag{15}$$

□

We can easily compute $-C_{41}$, $-C_{42}$ if C_{41}, C_{42} are known. By Theorem 1 or Lemma 1 to 4, we can compute X_4 with $-C_{41} \oplus -C_{42}$ instead of $C_{41} \oplus C_{42}$ because of C_{41}, C_{42} are odd numbers in the Yang et al. scheme. We can also easily obtain the keystream using only two pairs of plaintext and ciphertext. The verification of computation are listed as follows:

$$\begin{aligned} C_{41} &= 2FEFDB7Fh. \\ C_{42} &= FF270491h. \\ C_{41} \oplus C_{42} &= D0C8DFEEh. \\ -C_{41} &= FFFFFFFFD0102481h. \\ -C_{42} &= FFFFFFFF00D8FB6Fh. \\ -C_{41} \oplus -C_{42} &= D0C8DFEEh. \end{aligned} \tag{16}$$

4. Conclusion

The formal verification is one way to use mathematical methods to prove that scheme is correct or incorrect. A formal proof can ensure whether the result of logical inference is consistent with the previous stage, and can not guarantee whether there are defects in the process of logical inference. In this article the authors take as an example of XOR problem in Yu-Cao scheme, and then using two ways formal proofs; one is Boolean algebra and other is Galois field. Although the formal verification does not guarantee one hundred percent whether there are errors in logical inferences (such as tautology). However, at least in the process phase or the result phase of inference, it plays an important decisive role. It is very difficult to find or check the problem from mathematics or informatics fields. The authors fully use 2-adic number of Galois field to present this situation on Yu-Cao's scheme.

Acknowledgments

The authors would like to thank the reviewers for their comments that help improve the manuscript. This work is partially supported by the project number X2021110650493 from College Students Innovation and Entrepreneurship Training Program of China by Qingdao University.

References

[1] X. Wang and J. Zhao, "An improved key agreement protocol based on chaos," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 12, pp. 4052–4057, 2010.

[2] W. Yu and J. Cao, "Cryptography based on delayed chaotic neural networks," *Physics Letters A*, vol. 356, no. 4, pp. 333–338, 2006.

[3] J. Yang, X. Liao, W. Yu, K. wo Wong, and J. Wei, "Cryptanalysis of a cryptographic scheme based on delayed chaotic neural networks," *Chaos, Solitons and Fractals*, vol. 40, no. 2, pp. 821–825, 2009.

[4] C. Liu, S. Chen, and S. Sun, "Security of analysis mutual authentication and key exchange for low power wireless communications," *Energy Procedia*, vol. 17, pp. 644–649, 2012.

[5] X. Ye, X. Ye, and R. Wu, "Security analysis of Yu-Cao neural networks scheme," in *AIP Conference Proceedings*, vol. 2040, 2018, pp. 1 300 061–1 300 065.

[6] A. Rogers, J. G. Keating, R. Shorten, and D. M. Heffernan, "Chaotic maps and pattern recognition-the XOR problem," *Chaos, Solitons and Fractals*, vol. 14, pp. 57–70, 2002.
[7] S. Xu, Y. Wang, J. Wang, and M. Tian, "Cryptanalysis of two chaotic image encryption schemes based on permutation and XOR operations," in *2008 International Conference on Computational Intelligence and Security*, vol. 2, 2008, pp. 433–437.
[8] X. Wang, N. G. H. Zhao, S. Wang, and Y. Zhang, "A new image encryption scheme based on coupling map lattices with mixed multi-chaos," *Scientific Reports*, vol. 10, no. 1, pp. 1–15, 2020.
[9] M. I. Garcia-Planas, "Analyzing controllability of neural networks," *WSEAS Transactions on Circuits and Systems*, vol. 18, pp. 1–6, 2019.
[10] L. Kanaan, J. Haydar, M. Samaha, A. Mokdad, and W. Fahs, "Intelligent bus application for smart city based on LoRa technology and RBF neural network," *WSEAS Transactions on Systems and Control*, vol. 15, pp. 725–732, 2020.
[11] W. Yang, Y. Chai, J. Zheng, and J. Liu, "Intelligent diagnosis technology of wind turbine drive system based on neural network," *WSEAS Transactions on Circuits and Systems*, vol. 19, pp. 289–296, 2020.
[12] S. C.-I. Chen, C. Liu, Z. Wang, R. McAdam, M. Brennan, S. Davey, and T. Y. Cheng, "How geographical isolation and aging in place can be accommodated through connected health stakeholder management: Qualitative study with focus groups," *Journal of Medical Internet Research*, vol. 22, no. 5, p. e15976, May 2020.
[13] S. A. Al-Mawsawi, A. Haider, and Q. Alfaris, "Neural network model predictive control (NNMPC) design for UPFC," *WSEAS Transactions on Computers*, vol. 19, pp. 201–207, 2020.
[14] Z. Sultana, A. R. Khan, and N. Jahan, "Early breast cancer detection utilizing artif cial neural network," *WSEAS Transactions on Biology and Biomedicine*, vol. 18, pp. 32–42, 2021.
[15] C. Voloşencu, "Study of the angular positioning of a rotating object with neural predictive control," *WSEAS Transactions on Computers*, vol. 20, pp. 234–238, 2021.
[16] T. Xiang, X. Liao, G. Tang, Y. Chen, and K.-W. Wong, "A novel block cryptosystem based on iterating a chaotic map," *Physics Letters A*, vol. 349, no. 1, pp. 109–115, 2006.
[17] Y. Wang, X. Liao, T. Xiang, K.-W. Wong, and D. Yang, "Cryptanalysis and improvement on a block cryptosystem based on iteration a chaotic map," *Physics Letters A*, vol. 363, no. 4, pp. 277–281, 2007.
[18] X. Wang and C. Yu, "Cryptanalysis and improvement on a cryptosystem based on a chaotic map," *Computers and Mathematics with Applications*, vol. 57, no. 3, pp. 476–482, 2009.
[19] C. Li, Y. Liu, L. Y. Zhang, and M. Z. Q. Chen, "Breaking a chaotic image encryption algorithm based on modulo addition and XOR operation," *International Journal of Bifurcation and Chaos*, vol. 23, no. 4, pp. 1–12, 2013.
[20] X. Wu, H. Hu, and B. Zhang, "Analyzing and improving a chaotic encryption method," *Chaos, Solitons and Fractals*, vol. 22, no. 2, pp. 367–373, 2004.