

On the Security of a Clipped Hopfield Neural Network-Based Cryptosystem

Daniel Socek and Dubravko Čulibrk
Dept. of Computer Science and Engineering
Center for Cryptology and Information Security
Florida Atlantic University
777 Glades Road
Boca Raton, Florida 33431-0991, USA
{dsocek,dculibrk}@fau.edu

January 29, 2008

Abstract

A cryptosystem based on a clipped Hopfield neural network (CHNN) was recently proposed primarily for encryption of digital images and videos. The system is fast and suitable for hardware implementation. The present paper investigates the security aspects of the CHNN-based cryptosystem, and the following weaknesses are pointed out: 1) the cryptosystem is not sufficiently secure against the ciphertext-only attacks due to the weak randomness properties of the generated keystream, and 2) the cryptosystem is insecure against known/chosen-plaintext attacks and only one known plaintext-ciphertext pair is enough to completely break all ciphertexts of the same or smaller size obtained using the same encryption keys. The security of CHNN-based cryptosystem cannot be improved unless the basic model is fundamentally changed.

1 Introduction

The security of multimedia is very important in the modern computerized and interconnected world. The confidentiality and protection of media content must be enforced in applications such as private multimedia messages, pay-per-view TV, confidential video conferencing, medical imaging, and in applications related to industrial or military multimedia systems. Encrypting an entire multimedia bitstream using a conventional cryptosystem such as AES is referred to as the *naïve approach*. Unfortunately, in many applications, applying conventional encryption algorithms on the entire bitstream is not suitable for image and video encryption [9, 10, 11]. For example, a limited bandwidth and processing power in small mobile devices call for different approaches. In that respect,

many fast algorithms specifically designed for encrypting a particular multimedia data have been proposed [1, 2, 3, 4, 5]. However, a number of these algorithms have been shown to be insecure [6, 7, 8].

Many encryption algorithms for digital images and videos are based on encrypting only certain parts of the bitstream in order to reduce the amount of computation. This is known as the *selective encryption*. In addition, there are image and video encryption approaches based on fast chaotic maps that encrypt either the entire bitstream or a part of it. In [5], Chan et al. introduced a slightly different approach, where a *clipped Hopfield neural network* (CHNN) was used to generate a fast multimedia encryption scheme. However, this CHNN-based cryptosystem has weaknesses and it is subject to some well-defined cryptanalytic attacks. In this paper, we examine the weaknesses of a CHNN-based cryptosystem and present the relevant cryptanalytic attacks.

The paper is organized as follows. Section 2 briefly introduces the theory of clipped Hopfield neural networks, while the framework of a CHNN-based encryption algorithm from [5] is described in Section 3. Our main contribution is Section 4 which examines the security of the CHNN-based encryption algorithm and presents several applicable cryptanalytic attacks. The last section summarizes our conclusions.

2 The Clipped Hopfield Neural Network (CHNN)

In [13], Hopfield described a class of asynchronous neural networks which can be used as associative memory systems. These systems exhibit emergent properties that cause them to converge to one of the stable, low-energy states of the system regardless of the initial state of the system. These stable states of the system are called *attractors*, and the system can be engineered to have arbitrary stable states, representing the memory content. Hopfield neural networks (HNN) converge to the local stable state of the system closest to the system's initial state, exhibiting stochastic error in the process. For a given initial state (input of the HNN), HNN will therefore produce an output (attractor of HNN) that cannot be determined based only on the knowledge of the input. A CHNN represents a specific class of HNN, in which the synaptic weight matrix is clipped to the following three values: $\{-1,0,1\}$.

The CHNN used in [5] has been engineered to memorize $2N + 1$ attractors, where N is the number of neurons in the CHNN. This choice warrants some explanation. The theoretical upper bound on the capacity of such a CHNN is $2N$. Moreover, if we require the network to be able to hold every single set of attractors of certain dimensions, the capacity of the network is N [14]. Indeed, the simulation of the CHNN proposed in [5] showed that the network is able to memorize only N attractors. For the purpose of simulation, the number of nodes N in the CHNN was 8, a value inferred based on the description of proposed cryptosystem. The reduced number of memorized attractors did not affect the balancedness of the output, and consequently the output remained balanced.

Let $S_i(t)$ denote the state of the i -th neuron of the network at time t and

T_{ij} the synaptic weight of the connection between i -th and j -th neuron. The next state of each neuron $S_i(t+1)$ is determined by the current states of other neurons in the following way:

$$S_i(t+1) = f\left(\sum_{j=0}^{N-1} T_{ij}S_j(t)\right),$$

where $f(\cdot)$ is a nonlinear function:

$$f(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0. \end{cases}$$

This can be written in a more concise way using the matrix notation:

$$S(t+1) = f(S(t)T) = F_T(S(t)) = F_T^t(S(0)) \quad (1)$$

where $S(t+1) = S_i(t+1), i = 1, \dots, N$ and $f(x)$ is obtained by applying the f on each of the elements in x . T is the matrix of synaptic weights.

The asynchronous character of the HNN, in digital systems, translates usually to randomly selecting a neuron to propagate with the same probability of selection for each neuron. One should note that if the sequence in which the neurons are triggered is kept the same, a HNN will produce the same output each time a certain initial state is applied as input. If it were possible to implement a truly random selection of the neuron to trigger next, the HNN would exhibit truly random properties which would make it unsuitable for use in cryptosystems. Finally, if a pseudo-random number generator is used to determine the sequence in which the neurons are triggered, the HNN will generate a sequence of the complexity and the periodicity of the used generator.

3 The CHNN-Based Cryptosystem

In [5], Chan et al. proposed a novel cryptosystem based on clipped Hopfield neural networks. The proposed cryptosystem is here simply referred to as CHNNC. In essence, CHNNC acts as a stream cipher. The encryption procedure of CHNNC is realized as follows.

Let K_1 and K_2 be two secret 16-bit keys, and let $K_i[a:b]$, $i \in \{1, 2\}$, denote a segment of K_i that starts from the more significant bit at position a and terminates with the less significant bit at position b . In such notation, $K_1[15:0]$ denotes the entire key K_1 . Furthermore, if we equally partition the set of attractors into four bins denoted by Ψ_0 , Ψ_1 , Ψ_2 , and Ψ_3 , let $O(\cdot)$ denote an output mapping function defined as follows:

$$O(x) = \begin{cases} 00, & \text{if } x \in \Psi_0; \\ 01, & \text{if } x \in \Psi_1; \\ 10, & \text{if } x \in \Psi_2; \\ 11, & \text{if } x \in \Psi_3. \end{cases}$$

Finally, let \parallel be the usual concatenation and let C_P denote a clipped Hopfield neural network with eight attractors ($N = 8$), which is permuted according to a permutation P . For the input state x , the output state of C_P is represented by $C_P(x)$. Then the i -th byte B_i of the keystream is defined as $B_i = b_i^4 \parallel b_i^3 \parallel b_i^2 \parallel b_i^1$, and each b_i^j , $j = 1, 2, 3, 4$, is defined as follows:

$$b_i^j = O(C_{P_i^j}(K_1[(4j-1):(4j-4)] \parallel K_2[(4j-1):(4j-4)])).$$

Here, each permutation P_i^j is assumed to be generated using some fast pseudo-random permutation generator (PRPG) of limited complexity and low periodicity. The unit of four neural networks that generates a keystream byte is referred to as a CHNN *cell*. A keystream byte generated from a CHNN cell is then XOR-ed with the plaintext byte to obtain the ciphertext byte. At the other end, decryption is simply realized by XOR-ing the same keystream with the ciphertext.

Even though in the original proposal each of the two keys is split into 4-bit segments that are interchangeably applied, it is rather easy to observe that such key segmentation does not affect the security of the algorithm whatsoever. Thus, we shall simply refer to the 32-bit key used with a CHNN cell as K .

4 Cryptanalysis of the CHNN-Based Cryptosystem

The advantages of the CHNNC scheme over the conventional symmetric-key cryptosystems such as DES is the fast performance and suitability for hardware realization. In [5] the authors of CHNNC proposed a simple VHDL hardware design for the scheme, however the PRPG engine was not specified for such setting. In this section we show that the apparent advantages of CHNNC are unfortunately achieved at the expense of security.

4.1 Ciphertext-Only Attacks

In a ciphertext-only attack, the adversary only has an access to one or more encrypted messages. The most important goal of a proposed cryptosystem is to withstand this type of attack. An important instance of ciphertext-only attack is the brute-force attack. This type of attack is based on an exhaustive key search, and for a well-designed cryptosystem, it should be computationally infeasible.

In the case of the CHNNC system, the complexity of the brute-force attack directly depends of the chosen CHNNC setup. If n denotes the number of chosen CHNN cells, then the size of the input secret key is $n \times 32$ bits. In addition, the pseudo-random permutation generator must be initialized with a secret seed. If we assume that PRPG takes an initial key of size m bits, then the total key space of CHNNC is 2^{32n+m} . Thus, the values of parameters m and n directly affect the feasibility of the brute-force attack based on the exhaustive key search.

In modern symmetric-key cryptography the lowest recommended key size is 64 bits (DES), however the use of longer keys with sizes 128, 192, and 256 bits is encouraged (AES). Therefore, one should choose m and n so that $32n + m \geq 64$.

Unfortunately, CHNNC is a subject to a more efficient cipher-text-only attack due to a serious security flaw. The approach described in the original paper is based on the belief that the energy of a CHNN can be changed significantly through small changes in the weight matrix parameters. However, the authors attempt to introduce such changes by applying a random permutation to the weight matrix. They apply the permutation to the attractors ξ_i ($i = 0, 1, \dots, 2N$) and the weight matrix T as follows:

$$\widehat{\xi}_i = \xi_i P^T, \quad i = 0, 1, \dots, 2N \quad (2)$$

and

$$\widehat{T} = PTP^T \quad (3)$$

where P is a permutation matrix and P^T is the transpose (inverse) of P . The authors of the original approach claim that for the same input imposed on the CHNN, it will converge to different attractor for different P . By changing P , the authors propose to achieve different sequences.

The transformation described by equations (2) and (3) is actually a renaming of the neurons of the HNN by permuting their indices. In fact, \widehat{T} is obtained by conjugating T with the permutation P , i.e., first the rows of T are permuted according to P and then the elements of each row are permuted according to P . Such an operation does not change the static structure of the network nor does it affect its dynamic behavior. Hence, the energy of the system remains unchanged when the indices of the neurons are permuted, which was mathematically proven in [15]. Next, we prove the following key theorem:

Theorem 1. *Let P is an $N \times N$ permutation matrix, and let T and \widehat{T} be two $N \times N$ synaptic matrices of a CHNN, such that $\widehat{T} = PTP^T$. If $S(0)$ denotes the initial state of the network, then*

$$F_{\widehat{T}}^t(S(0)) = F_T^t(S(0)P)P^T$$

holds for all $t \geq 1$.

Proof. We perform an induction on t . It is easy to see that the equation holds for $t = 1$:

$$\begin{aligned} F_{\widehat{T}}^1(S(0)) &= f(S(0)PTP^T) \\ &= f(S(0)PT)P^T \\ &= F_T^1(S(0)P)P^T. \end{aligned}$$

Note here that we could put P^T outside of the clipping function f since the multiplication on the right by P^T is equivalent to permutating the elements of each row of the matrix on the left according to the permutation P . Assuming

that the theorem holds for $t - 1$, we have the following equality:

$$\begin{aligned}
F_{\widehat{T}}^t(S(0)) &= f(F_{\widehat{T}}^{t-1}(S(0))\widehat{T}) \\
&= f(F_{\widehat{T}}^{t-1}(S(0))PTP^T) \\
&= f(F_T^{t-1}(S(0)P)P^TPTP^T) \\
&= f(F_T^{t-1}(S(0)P)T)P^T \\
&= F_T^t(S(0)P)P^T.
\end{aligned}$$

Hence, the theorem holds for all $t \geq 1$ by induction. \square

In other words, Theorem 1 states that the output of the permuted network is identical to the result obtained by permuting the input to the regular network according to P and then permuting the network output according to the inverse of P , which is a far cheaper procedure than permuting the network itself. Moreover, this reveals that the function of the network is just a nonlinear mapping and can in no way alter the periodicity of the pseudo-random number generator employed to create the pseudo-random permutations.

As a comment on the results the authors report in [15], that in some cases some of the inputs of the permuted HNN converge to attractors different than permuted attractors to which they converge in the original HNN, there can be only one explanation for the observed behavior. The error of a HNN is stochastic and the network will in some cases converge to different attractors, depending on the sequence in which the neurons of the network are triggered. Although the authors do not give details of the concrete implementation of the HNN they used, it is likely that they did not take care to permute the sequence in which they triggered the neurons, along with permuting the HNN, or that they used a pseudo-random generator to determine the triggering sequence. In the latter case, the only viable choice in terms of computational cost would be to use the same pseudo-random generator used to create the permutation matrix P . Clearly the periodicity of the whole system corresponds to that of the pseudo-random generator used.

It is duly noted here that in an ideal HNN the neurons are triggered randomly, making the behavior (i.e., the error) of the HNN truly random. This non-deterministic characteristic would make such HNNs unsuitable for use in cryptosystems.

Since applying the pseudo-random permutation to the synaptic weight matrix T is equivalent to permuting the input key K , a CHNN cell can be thought of as a black-boxed surjective keyed hash function that deterministically maps a 32-bit input into an 8-bit output. Without loss of generality assume that only one CHNN cell is used ($n = 1$). If the plaintext consists of k bytes, the black-box is iterated k times to obtain k bytes of the keystream. In each iteration, K is permuted according to the pseudo-random permutation. Therefore, the randomness of the hashed output directly depends on the randomness of the permutation generator. But in CHNNC, each permutation is generated using a weak PRPG with low periodicity and complexity. Thus, the keystream is a bit sequence of low periodicity that is subject to a well-defined statistical

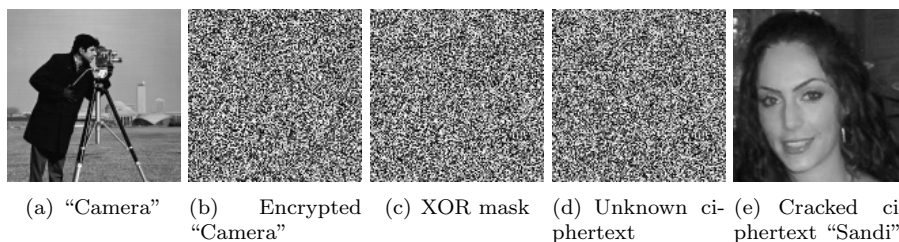


Figure 1: Chosen/known-ciphertext attack on CHNNC: the attacker calculates (c) by XOR-ing (a) and (b), and then (e) by XOR-ing (c) and (d).

cryptanalysis, such as Kasiski analysis that is effective against the Vigenère-type ciphers. The scheme is only improved if a strong random number/permutation generator is used, such as the one proposed in [12], but the efficiency of the scheme is then sacrificed. In addition, it follows that it is more efficient to directly use the underlying pseudo-random number generator as a keystream, and completely bypass the neural network stage.

4.2 Known/Chosen-Plaintext Attacks

In a known-plaintext attack, the adversary has some knowledge about the plaintext corresponding to the given ciphertext. This may help determine the key or a part of the key, which ultimately could result in recovering the entire set of messages that were encrypted using that particular key. In the chosen-plaintext attack, the adversary can feed the chosen plaintext into the black-box that contains the encryption algorithm and the encryption key. The black-box spits out the corresponding ciphertext, and the adversary may use the accumulated knowledge about the plaintext-ciphertext pairs to obtain the secret key or at least a part of it.

A cryptosystem that is susceptible to known/chosen-ciphertext attacks is not recommended in general. It is especially not recommended for multimedia security systems. Having to change the key from image to image, or frame to frame is a big drawback for many applications. In addition, numerous videos are likely to contain predictable frames such black introductory frames, MPAA ratings, FBI warnings, MGM roaring lion sequences, or other usual jingles. In that respect, ciphers that are vulnerable to known-plaintext attacks are not favorable.

In [5] (Sec. 5), the authors of CHNNC claimed that neither a chosen-plaintext attack nor a known-plaintext attack could be used to find the secret key. Although it is not clear how to recover the original secret key K , recovering the corresponding keystream is just as effective in breaking the system. Under the known/chosen-plaintext attack, only one known plaintext-ciphertext pair is enough to completely break all ciphertexts of same or smaller size obtained using the same encryption keys. Since the encryption is realized as a simple XOR-

ing of the plaintext with the keystream, XOR-ing the plaintext and ciphertext pair results in the keystream due to the fact that XOR is an involution. The known/chosen-plaintext attack on CHNNC can be best illustrated using the encryption of images. Suppose we have the images I and its CHNNC encryption I' obtained by using secret key K . By virtue of the algorithm's definition, I' can be obtained from I by XOR-ing it with a particular image mask I_m . Consequently, the image mask I_m can be obtained simply by XOR-ing images I and I' . This mask can then be used to completely decrypt all other images of same or smaller size encrypted with the same key K . Fig. 1 demonstrates chosen/known-plaintext attack on CHNNC where the same key is used to encrypt both 128×128 greyscale image "Camera" and 128×128 greyscale image "Sandi". The attacker can easily recover "Sandi" from the unknown ciphertext in Fig. 1(d).

The attack was possible because of the following property. Let $p = I(i, j)$ be a pixel value of I at coordinates (i, j) . Then $p' = I'(i, j)$ can be expressed as $p' = p \oplus x$, for some keystream byte x . By the framework of CHNNC, $q' = J'(i, j) = q \oplus x$, where $q = J(i, j)$. If q is unknown and p, p' and q' are known, then q is easily calculated as follows:

$$\begin{aligned} p \oplus p' \oplus q' &= p \oplus (p \oplus x) \oplus (q \oplus x) = \\ &= p \oplus p \oplus x \oplus x \oplus q = q. \end{aligned}$$

In general, all cryptosystems that transform the plaintext solely by using an XOR mask suffer from the same problem and as such cannot resist the known/chosen-plaintext attacks. To resist such attacks one must ensure both confusion and diffusion of the data, but this can only be accomplished by a completely redesigned cryptosystem.

5 Conclusions

In this paper, we investigated the security of a recently proposed cryptosystem based on clipped Hopfield neural networks [5]. It is shown that CHNNC is insecure against the ciphertext-only attacks and the known/chosen-plaintext attacks. The susceptibility of CHNNC to known/chosen-plaintext attacks is due to using a simple XOR transformation of the plaintext, and such weakness is common to all cryptosystems that are solely based on the XOR masking. Furthermore, the ciphertext-only attacks are possible since the CHNNC transformation does not improve the randomness of the generated keystream due to the properties of a deterministic clipped Hopfield neural network. As a conclusion, CHNNC is not recommended for use in any serious security applications.

6 Acknowledgments

The authors would like to thank Prof. Spyros S. Magliveras at the Center for Cryptology and Information Security and Mathematical Sciences at Florida

Atlantic University for taking time to review the manuscript of this paper. This research was partially supported by a Federal Earmark grant for *Research in Secure Telecommunication Networks* (2004-05).

References

- [1] L. Tang. Methods for encrypting and decrypting MPEG video data efficiently. In *Proceedings of the 4th ACM International Multimedia Conference*, pages 219–230, 1996.
- [2] H. Cheng and X. Li. Partial encryption of compressed images and video. In *IEEE Transactions on Signal Processing*, volume 48, pages 2439–2451, 2000.
- [3] Jui-Cheng Yen and Jiun-In Guo. A new chaotic key-based design for image encryption and decryption. In *Proceedings of 2000 IEEE International Conference on Circuits and Systems (ISACS 2000)*, volume 4, pages 49–52, 2000.
- [4] B. Bhargava, C. Shi, S.-Y. Wang. MPEG Video Encryption Algorithms. *Multimedia Tools and Applications*, Kluwer Academic Publishers, Vol. 24, No. 1, pages 57–79, 2004.
- [5] C.-Ku. Chan, C.-Kw. Chan, L.-P. Lee, and L.M. Cheng. Encryption System Based On Neural Network. In *Proceedings of IFIP TC6/TC11 Fifth Joint Working Conference on Communications and Multimedia Security (CMS '01)*, Darmstadt, Germany, May 21-22 (2001), *Communications and Multimedia Security Issues of the New Century*, Kluwer Academic Publishers, Boston, MA, pages 117–122, 2001.
- [6] K. Nahrstedt, L. Qiao and I. Tam. Is MPEG encryption by using random list instead of zigzag order secure? In *IEEE International Symposium on Consumer Electronics*, Singapore, 1997.
- [7] T. Seidel, D. Socek and M. Sramka. Cryptanalysis of video encryption algorithms. In *Proceedings of Third Central European Conference on Cryptology (TATRACRYPT 2003)*, Bratislava, Slovak Republic, June 26-28 (2003), Tatra Mountains Mathematical Publications, volume 29, pages 1–9, 2004.
- [8] Shujun Li and Xuan Zheng. Cryptanalysis of a chaotic image encryption method. In *Proceedings of 2002 IEEE International Symposium on Circuits and Systems (ISCAS 2002)*, volume 2, pages 708–711, 2002.

- [9] B. Furht and D. Socek. Multimedia security: Encryption techniques. In *IEC Comprehensive Report on Information Security, International Engineering Consortium*, Chicago, 2003.
- [10] Shujun Li, Guanrong Chen and Xuan Zheng. *Multimedia Security Handbook* edited by B. Furht and D. Kirovski, volume 4 of *Internet and Communications Series*, chapter “Chaos-Based Encryption for Digital Images and Videos”. CRC Press, December 2004.
- [11] Borko Furht, Daniel Socek, and Ahmet M. Eskicioglu. *Multimedia Security Handbook* edited by B. Furht and D. Kirovski, volume 4 of *Internet and Communications Series*, chapter “Fundamentals of Multimedia Encryption Techniques”. CRC Press, December 2004.
- [12] S.S. Magliveras and N.D. Memon. Random permutations from logarithmic signatures. In *Computing in the 90's - First Great Lakes Computer Science Conference*, volume 507 of *Lecture Notes in Computer Science*, pages 91–97. Springer-Verlag, Berlin, 1989.
- [13] J.J. Hopfield. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. In *Proc. Natl. Acad. Sci.*, volume 79, pages 2554–2558, 1982.
- [14] R.J. McEliece, E.C. Posner, E.R. Rodemich, S.S. Vankatesh. The Capacity of the Hopfield Associative Memory. In *IEEE Trans. Inform. Theory*, IT-33(4), pages 461–482, 1987.
- [15] D. Guo, L.M. Cheng, L.L. Cheng, A New Symmetric Probabilistic Encryption Scheme Based on Chaotic Attractors of Neural Networks. In *Applied Intelligence*, 10(1), 1999.