Cryptanalysis of a cryptosystem based on discretized two-dimensional chaotic maps

Ercan Solak*, Cahit Çokal

Department of Computer Engineering, Isik University, Istanbul, Turkey

A R T I C L E   I N F O

Article history:
Received 12 August 2008
Received in revised form 14 September 2008
Accepted 9 October 2008
Available online 15 October 2008
Communicated by A.R. Bishop

PACS:
05.45.Vx

Keywords:
Communication using chaos
Cryptanalysis

A B S T R A C T

Recently, an encryption algorithm based on two-dimensional discretized chaotic maps was proposed [Xiang et al., Phys. Lett. A 364 (2007) 252]. In this Letter, we analyze the security weaknesses of the proposal. Using the algebraic dependencies among system parameters, we show that its effective key space can be shrunk. We demonstrate a chosen-ciphertext attack that reveals a portion of the key.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

In this Letter, we cryptanalyze the chaotic encryption algorithm proposed in [1]. The algorithm uses discretized two-dimensional chaotic maps (TDCM) and S-boxes designed using chaotic systems. We first show that the key contains redundancies that lead to a shorter effective key length. Next, we demonstrate a chosen-ciphertext attack to recover a portion of the key.

2. Description of the algorithm

The encryption algorithm processes a sequence of 16-bit plaintext blocks and produces another sequence of 16-bit ciphertext blocks. Plaintext and ciphertext sequences are partitioned into 16-bit blocks $P_i$, $1 \leq i \leq n$, as

 Plaintext: $P_1P_2\cdots P_n$.

 Ciphertext: $C_1C_2\cdots C_n$.

The key of the cryptosystem is the collection of the parameters $(r, m, t, C_0, K_s, K_c)$. In [1] this collection is defined as the master key. The master key is composed of the number of rounds $r$, the shift amount $m$, the number of iterations $t$, the initial value $C_0$, the subkey $K_s$ and the collection of TDCM parameters $K_c$.

A block key $K_i$ is used in the encryption of plaintext block $P_i$. Initially, we have

$$K_0 = K_s.$$  \hfill (1)

Before the encryption of block $P_i$, $K_i$ is first updated as

$$K_i = \begin{cases} 
K_{i-1} \oplus C_{i-1} & \text{if } C_{i-1} \neq K_{i-1}, \\
K_{i-1} & \text{if } C_{i-1} = K_{i-1}.
\end{cases}$$ \hfill (2)

The encryption of the $i$th block is given as

$$C_i = E(K_i, P_i),$$ \hfill (3)

where the function $E$ involves the following round operations:

$$v_0 = P_i,$n

$$v_j = \sigma(v_{j-1} \oplus \text{ROL}(K_i, jm)), \quad 1 \leq j \leq r,$n

$$C_i = v_r.$n

Here, $v_j$ is the output of round $j$. \text{ROL}(\cdot, jm)$ denotes the circular left rotation by $jm$ bits. The round function $\sigma$ is given as

$$\sigma = w \circ z^{-1} \circ \text{TDCM}_{K_c} \circ z \circ S.$$ \hfill (5)

The amount of circular left shifts is given as

$$m = \begin{cases} 
16/r, & r \leq 16, \\
1 & \text{otherwise}.
\end{cases}$$ \hfill (6)

In (5), $S$ represents the S-box substitution. $S$ invertibly maps between 16-bit quantities. The S-box is designed to have desirable

---

* Corresponding author. Tel.: +90 216 5287149; fax: +90 216 7102872.
E-mail addresses: ercan@isikun.edu.tr (E. Solak), cahit.cokal@isik.edu.tr (C. Çokal).
nonlinear properties, and its value is fixed (not secret) for an algorithm. Ref. [2] gives examples of S-boxes designed using iterations of chaotic systems.

$z$ is an invertible function that maps from 16-bit quantities to 2D vectors of integers. It maps the unsigned integer values corresponding to each byte of its argument to one of the integer coordinates in 2D discrete state space. For example, $z(0x{\text{F3A7}}) = [243, 167]$ because $243 = 167 = (A^2)_{16}$.

$\text{TDCM}^{i}$ denotes the $i$-times iteration of TDCM. $K_c$ denotes the collection of the chaotic system parameters. The choice of the chaotic map is part of the algorithm design. In [1], the standard map, the generalized cat map, and the generalized baker map are considered. The chaotic map must be bijective in order to have an invertible encryption operation. The output of the chaotic system is passed through $z^{-1}$ to map the final 2D state of TDCM to a 16-bit number.

The last mapping $w$ in (5) denotes the byte swap operation.

After the encryption of block $i$, the block key is once more updated as $K_i = \text{ROL}(K_i, r_m)$.

Since $K_i$ is 16-bits, the effective amount of rotation on $K_i$ in this step is $r_m \mod 16$.

3. Key space weakness

The cryptosystem described in the previous section uses the secret parameters $r(8), m(8), t(8), C_0(16), K_r(16)$. The numbers between the brackets are the number of bits used to represent the parameter. The parameters $K_r$ of the TDCM also contribute to the key space. For example, the standard map has a single parameter which is represented using 16 bits. In this case, the master key has 72 ($56 + 16$) bits. Using a simple brute force, an attacker has to try $2^{71}$ keys on average until he finds the correct key.

However, algebraic dependencies present in the system make the effective key size smaller.

The relation (6) fixes $m$ once $r$ is known. This removes the freedom in the choice of $m$, and effectively reduces the key length by 8 bits. Therefore, the shift amount $m$ must be treated not as a key but rather as an internal parameter that is derived from the key.

Another reduction in effective key length is due to the way the secret parameter $C_0$ is used. Before the encryption of the first 16-bit block, the subkey $K_r$ is updated by using (2). Hence, the value of $K_1$ used in the encryption of $P_1$ is $K_r \oplus C_0$. Consequently, we can treat $K_r \oplus C_0$ as one secret parameter rather than two distinct parameters, $K_r$ and $C_0$. Indeed, any pair of $C_0$ and $K_r$ values that yields the same XOR value results in identical encryption functions.

This fact reduces the effective key length by another 16 bits. In the subsequent sections, we assume without loss of generality that $C_0 = 0x0000$.

One might remedy the key space weakness by using a larger $K_r$. However, as we show in the sequel, there are attacks that work whatever the size of $K_r$ is.

4. Chosen ciphertext attack on $K_r$

Assume that the attacker knows the number of rounds $r$. This is not a very restrictive assumption. Since $r$ is represented with 8 bits, it can only take one of 255 possible nonzero values. The attacks that we develop in this and the next section have very low computational requirements. In the case when the attacker does not know the value of $r$, he tries all 255 possible values with the attacks described here. He then eliminates false $r$ values by trying the encryption against a couple of known plaintext–ciphertext pairs. Namely, the attacker uses brute-force for recovering $r$, once he has fast methods to attack the rest of the key.

To illustrate the method of the attack, we first analyze the case when $r_m \equiv 0 \mod 16$. Later in the section we will give the attack that works when $r_m \not\equiv 0 \mod 16$.

We assume that the attacker does not know the TDCM parameters, so he does not know the function $E$ in (3).

4.1. $r_m \equiv 0 \mod 16$.

Assume that the first two ciphertext blocks are given as

$$C_1 = C_2 = J. \quad (8)$$

If $j = K_r$, using (1), (2), (3) and (7), we have

$$j = E(K_r, P_1), \quad j = E(K_r, P_2).$$

So, by the invertibility of $E$ for fixed $K_r$, we have

$$j = E(K_r, P_1), \quad j = E(K_r \oplus j, P_2).$$

In this case, most probably $P_1 \not= P_2$. The difference in two cases indicates that the equality of $P_1$ and $P_2$ is a good test on whether $K_r = j$.

The attack on $K_r$ proceeds as follows. The attacker chooses a 16-bit number $h$. He requests plaintexts for a two-block ciphertext $C_1C_2$ chosen as in (8). He compares these plaintext blocks $P_1$ and $P_2$. If they are equal, then $j$ is a candidate for the secret $K_r$. The attacker repeats this for all the 16-bit $j$ values and records candidates for $K_r$. A total of $2^{16} - 1$ trials are made.

It may happen that the attacker obtains $P_1 = P_2$ even when $j \not= K_r$. This is because we might have $E(K_1, P) = E(K_2, P)$ for some $K_1 \not= K_2$, and $P$. In order to eliminate the false keys, the attacker performs the following further tests.

Assume that the attacker has two candidates $j_1$ and $j_2$ for the subkey $K_r$. From his previous attempt at determining the keys, the attacker knows $P_1$ and $P_2$ which satisfy

$$j_1 = E(K_r, P_1), \quad j_2 = E(K_r, P_2). \quad (9)$$

The attacker now chooses the new ciphertext blocks $\tilde{C}_1$ and $\tilde{C}_2$ as

$$\tilde{C}_1 = j_1 \quad \text{and} \quad \tilde{C}_2 = j_2.$$ He obtains the corresponding plaintext blocks $\tilde{P}_1$ and $\tilde{P}_2$. There are two cases for the validity of $j_1$. Let us see how $P_1$ and $P_2$ differ for each case.

Case 1. $j_1 = K_r$. Using (1), (2), (3) and (7), we find that

$$j_1 = E(K_r, \tilde{P}_1), \quad j_2 = E(K_r, \tilde{P}_2).$$

Comparing this with (9), we obtain $\tilde{P}_1 = P_1$ and $\tilde{P}_2 = P_2$.

Case 2. $j_1 \not= K_r$. This time we find,

$$j_1 = E(K_r, \tilde{P}_1), \quad j_2 = E(K_r \oplus j_1, \tilde{P}_2).$$

Comparing this with (9), we conclude $\tilde{P}_1 = P_1$ and $\tilde{P}_2$ is a random 16-bit number.

In both cases, $\tilde{P}_1 = P_1$. However, only in the first case we are guaranteed to have $P_2 = P_2$. In the second case, we might have $P_2 = P_2$ even when $j_1 \not= K_r$. So, if $P_2 = P_2$ the test is conclusive and $j_1 \not= K_r$. If $P_2 = P_2$ the test is inconclusive.

This test gives the attacker a method to eliminate the false subkeys among the candidates. Assume that attacker has determined $q$ candidates, $\{j_1, j_2, \ldots, j_q\}$ for the subkey $K_r$. To eliminate the false subkeys, he chooses a pair of candidates $j_i$ and $j_j$ and applies the test as explained. In this way, he eliminates $j_i$ if the test is conclusive. Otherwise, he chooses a different pair and repeats the test. The attack on $K_r$ successfully terminates when there remains only one candidate for the subkey.
4.2. $rm \not\equiv 0 \mod 16$

Let the ciphertext be chosen as
\[ C_1 = C_2 = \cdots = C_{k-1} = 0, \quad C_k = j, \quad C_{k+1} = 0, \quad (10) \]
where $k = \frac{km}{16 \cdot w}$ and $u = rm \mod 16$. Here, $k$ is chosen such that $K_1 = K_{k+1}$. Using (10) together with (1), (2), (3) and (7), we obtain
\[
0 = E(K_s, P_1),
0 = E(ROL(K_s, u), P_2),
0 = E(ROL(K_s, 2u), P_3),
\]
and
\[
0 = E(ROL(K_s, (k-2)u), P_{k-1}),
0 = E(K_s, (k-1)u), P_k),
(11)
\]
and
\[
0 = \begin{cases} E(j \oplus K_s, P_{k+1}) & \text{if } j \neq K_s, \\ E(K_s, P_{k+1}) & \text{if } j = K_s. \end{cases}
(12)
\]

Comparing (11) and (12), we find that if $j = K_s$, we have $P_1 = P_{k+1}$. The attacker uses this fact to launch a chosen ciphertext attack.

For a 16-bit nonzero number $j$, the attacker chooses the ciphertext sequence as in (10) and obtains the corresponding plaintext sequence $P_1, \ldots, P_{k+1}$. If $P_1 = P_{k+1}$, then $j$ is a candidate subkey. The attacker repeats this for all the 16-bit $j$ values and records candidates for $K_s$. A total of $2^{16} - 1$ trials are made.

It may happen with a low probability that we have $P_1 = P_{k+1}$ even when $j \neq K_s$. In order to rule out such a false key $j$, the attacker chooses the ciphertext sequence $C_1 = ROL(j, u)$, $C_2 = 0$ and obtains the corresponding plaintext sequence $P_1 P_2$. Using this with (1), (2), (3) and (7) we obtain
\[ ROL(j, u) = E(K_s, P_1), \]
and
\[
0 = \begin{cases} E(ROL(j, u) \oplus ROL(K_s, u), \bar{P}_2) & \text{if } j \neq K_s, \\ E(ROL(K_s, u), \bar{P}_2) & \text{if } j = K_s. \end{cases}
(13)
\]
Comparing (11) with (13), we see that if $j = K_s$, $P_2 = \bar{P}_2$. Thus, the attacker eliminates a false key $j$, if $P_2 \neq \bar{P}_2$.

In attacking the 16-bit subkey $K_s$, we used about $2^{10}$ chosen ciphertexts. It might seem that our attack is on the same order of a brute-force attack. However, when using our method, an attacker does not need to know the parameters $K_c$ and $t$, which characterizes the function $E$. In a brute-force attack, the attacker would need to know these parameters. A brute-force attack on a part of the key is possible only if the rest of the key is known.

5. Simulation results

We give simulation results for two examples that illustrate our methods.

In the first simulation, we used the cryptosystem with secret parameters given in the example in [1]. We have $r = 8$, $m = 2$, $t = 12$, $C_0 = 0x4ED3$, $K_s = 0x8F4C$. Using the equivalence explained in Section 3, this is equivalent to $C_0 = 0x0000$, $K_s = 0xC19F = 0x4ED3 \oplus 0x8F4C$. We used the standard map as TDCM. The secret TDCM parameter is $K_c = 53246$.

We applied the chosen ciphertext attack on $K_s$ given in Section 4. We found only one nonzero candidate for $K_s$. Hence we do not have false candidates for the subkey.

In the second example, we choose $r = 5$ and $m = 3$. The rest of the parameters are the same.

We apply the attack on $K_s$ for the case $rm \not\equiv 0 \mod 16$ given in Section 4.2. In this case, we have $u = 15$, so $K_s$ is rotated left by 15 bits after the encryption of every block of plaintext. We found two nonzero candidates for $K_s$: $0xC19F$ and $0xCFE1$. Using the elimination method given at the end of Section 4.2, we arrived at the correct subkey $K_s = 0xC19F$.

6. Conclusion

In this Letter, we gave a partial break of a cryptosystem that uses discretized two-dimensional chaotic maps. We showed a dependence among secret parameters that yield a smaller key space. We next showed that 16 bits of the key can be revealed using a chosen ciphertext attack. Using simulation with different parameters, we also demonstrated the feasibility of our attacks.

Acknowledgements

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under Project No. 106E143.

References