

# Data Encryption based upon Time Reversal Transformations

K. W. YU† AND T. L. YU\*‡

† Department of Applied Science, Hong Kong Polytechnic, Hung Hom, Hong Kong

‡ Department of Computer Science, California State University, San Bernardino, CA 92407, USA

*We present here a data encryption model based upon time reversal transformations in analogy to a mechanical system whose dynamics is described by second order (in time) equations. Such a mechanical system is conservative (i.e. no loss in energy). By the same token, if a set of data is manipulated according to an equation which is second order in time, the information contained in the data will not be lost after an arbitrary number of iterations. In this way, one can encrypt confidential data which is to be transmitted via a computer network to an incomprehensible form; the destined receiver can easily convert the encrypted data back to the original comprehensible form by reversing the order of iterations.*

Received October 1987, revised January 1988

## 1. INTRODUCTION

With the advent of computer and networking technologies a tremendous amount of information is moved between millions of computer terminals in the world every hour. Regardless of the types of networks that interconnect the terminals, the transmitting data can easily be copied by wiretap or other appropriate means; if a picture is sent via a satellite, it is accessible to anyone who has a TV set and an appropriate antenna. In many cases, the information stored in a computer is highly confidential and the organisations definitely want to prevent the information from being copied when it is transmitted through a computer network. In such cases, one may need to perform some form of encoding on the data prior to its transmission, so that it is incomprehensible to anyone except the sender and the destined receiver who knows how to decode the transmitting message. The coding operations are known as encryption (or encipherment) and the decoding operation decryption (or decipherment). The original data prior to being encrypted is referred to as plaintext and the encrypted data ciphertext.

A number of algorithms for encryption for computer data communication have been proposed in the past two decades.<sup>1-5</sup> However, the algorithms known to date are either very complicated – which can easily confuse even the sender and the receiver – or there may exist a danger that the associated ciphertext may be decoded by outsiders. Here we propose a very simple and straightforward encryption which can easily be implemented using either software or hardware or both; the ciphertexts are too complicated to be decrypted by someone who does not know the associated function-key. Moreover, the abundance of choices of function-key discussed in the next section makes ‘code-breaking’ impossible. Other advantages of the proposed encryption are discussed in Section 4.

In the next section, we describe in detail the model and the principles for encryption. Simple examples that highlight the model are presented in Section 3. We then discuss other properties and advantages of the model in Section 4. Section 5 summarises the important results.

## 2. THE MODEL AND BASIC EQUATIONS

In the physical world, if a system is described by an equation which is first-order in time, the system is in general dissipative (i.e. has energy loss). If the equation is second-order in time, the system may be non-dissipative. Such a system has time-reversal symmetry.<sup>6</sup> As an example in mechanics, if a certain initial configuration of a system of particles evolves under the action of various forces into some final configuration, a possible state of motion of the system is that the time-reversed final configuration (all positions the same, but all velocities reversed) will evolve over the reversed path to the time-reversed initial configuration. Therefore, if one wants to iterate a set of data for a number of times without any loss in information, so that when the iteration is reversed the original set of data can be recovered, the equation describing the iteration must be second-order in time. Based on this idea we develop the following model.

Suppose we have some data (information) arranged in an  $m \times n$  matrix  $(x_{ij})$ , ( $1 \leq i \leq m$  and  $1 \leq j \leq n$ ), each of which takes on  $k$  discrete values. The major purpose of encryption is to transform  $(x_{ij})$  into a new matrix  $(x_{ij}^*)$  by a known function  $f$  (henceforth referred to as the key).<sup>7</sup> In this way the transformed data  $(x_{ij}^*)$  is unintelligible. The basic equation of our transformation is

$$x_{ij}(T+1) = f[\{x_{ij}(T)\}] - x_{ij}(T-1) \pmod{k}. \quad (1)$$

In this equation,  $T$  labels the  $T$ th ( $T = 0, 1, 2, \dots$ ) copy of the sequence  $x_{ij}$ , namely,  $x_{ij}(0)$ ,  $x_{ij}(1)$ , ...,  $x_{ij}(T)$ , ... The key  $f[\{x_{ij}\}]$  which determines the transformed data  $(x_{ij}^*)$  is a function of  $x_{ij}$  and its neighbouring matrix elements. For a square matrix, if only nearest neighbours and  $x_{ij}$  itself are considered there are usually nine such matrix elements. It is obvious that if only these nine elements are utilised in the construction of the function, the total number of distinct functions is  $k$  to the power  $M$ , where  $M$  is  $k$  to the power 9; this is an astronomical number even when  $k = 2$ . (The existence of the enormous number of functions follows directly from the definition of functions.<sup>8</sup> However, for the completeness of this paper and to let the readers gain a perception of the abundance of the number of distinct functions we have included an appendix to discuss this in more detail.)

\* To whom correspondence should be addressed.

I encrypt thus I am!

>[ W t( ! ~ "Q+

b6Lb =h Qr

M >u' VM P b [A

pQ 7 f/u(4 i

L2Z+ 'C . c4 D

M >u' VM P b [A

b6Lb =h Qr

>[ W t( ! ~ "Q+

I encrypt thus I am!

Figure 1.

A simple example of  $f$  is the totality function, which involves a sum over the eight nearest neighbours of  $x_{ij}$  and  $x_{ij}$  itself:

$$f[\{x_{ij}\}] = x_{i-1,j-1} + x_{i,j-1} + x_{i+1,j-1} + x_{i-1,j} + x_{i,j} + x_{i+1,j} + x_{i-1,j+1} + x_{i,j+1} + x_{i+1,j+1}. \quad (2)$$

In practice, a finite sample is described by the matrix  $(x_{ij})$  with finite values of  $m$  and  $n$ ; we have to use the periodic boundary conditions

$$x_{i+m,j+n} = x_{ij}. \quad (3)$$

Equation (1) can be iterated for an arbitrary number of time steps provided two initial conditions are given:

$$x_{ij}(0) = b_{ij}, \quad x_{ij}(1) = c_{ij}. \quad (4)$$

In the model,  $b_{ij}$  can be an arbitrary background (which of course can also be the message that we want to send) and  $c_{ij}$  are the data to be encrypted. One of the most important properties of equation (1) is that it has time-reversal invariance; it can be rewritten in the following form:

$$x_{ij}(T-1) = f[\{x_{ij}(T)\}] - x_{ij}(T+1). \quad (5)$$

If two copies of the sequence  $\{x_{ij}\}$  are known at two successive time steps  $T$  and  $T+1$ , then the initial data  $c_{ij}$  can be recovered upon repeated iterations of equation (5).

We have described the two-dimensional case; one can easily reduce it to a one-dimensional

problem or generalise it to a three-dimensional one.

### 3. EXAMPLES

As an illustration of the model, three simple examples are presented as follows (Figs 1–3). All the printouts are from an IBM-PC system with colour graphics.

#### 3.1 One-dimensional message with 256 states ( $k = 256$ )

Here we consider messages which consist of ASCII characters. Each  $x_i$  has totally 256 states as there are 256 generalised ASCII characters; the value of  $x_i$  is equal to the ASCII code (0–255) representing the corresponding character.

Fig. 1 presents an example of such a case. The background values of  $x_i$  are set to be zero. The plaintext is 'I encrypt thus I am!'. The function key that has been used is

$$f[\{x_i\}] = 3x_{i-1} + 5x_i + 7x_{i+1}. \quad (6)$$

(Note that in general  $f$  is not necessarily linear with  $\{x_i\}$  though we have chosen a linear one; see Appendix.) In Fig. 1, line 1 is the plaintext; lines 2–6 are the ciphertexts in sequence. From the seventh line on, the ciphertexts are obtained by reversing the timing steps (i.e. line 6 becomes the background and line 5 becomes the data); line 10

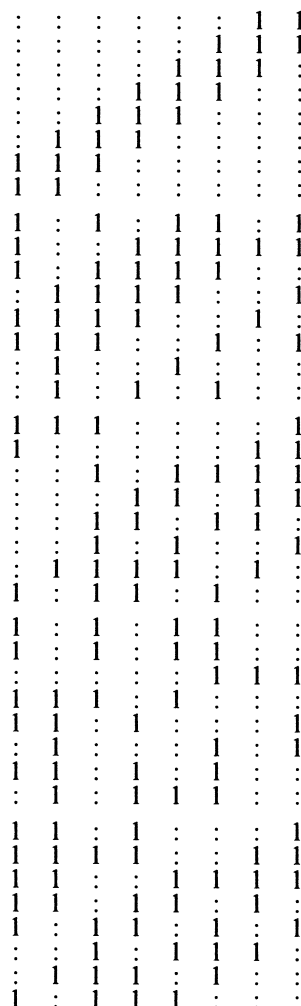


Figure 2.

:	:	:	:	:	:	1	1
:	:	:	:	:	:	1	1
:	:	:	:	:	:	1	1
:	:	:	:	:	:	1	:
:	:	:	1	1	1	:	:
:	:	1	1	1	:	:	:
:	1	1	1	:	:	:	:
1	1	1	:	:	:	:	:
1	1	:	:	:	:	:	:
3	2	1	:	1	3	:	3
1	:	:	1	3	1	1	3
1	:	1	3	1	1	:	2
:	1	3	1	1	:	2	1
1	3	1	1	:	2	1	:
:	1	:	2	1	:	:	2
:	3	2	1	:	1	2	:
1	3	3	:	2	:	2	1
3	:	:	2	2	2	3	1
2	2	1	2	3	3	1	1
:	2	2	3	3	2	3	1
2	2	3	3	2	3	1	:
:	2	3	2	3	:	2	3
:	1	1	3	1	2	3	2
1	:	3	1	:	3	:	:
1	:	1	2	1	1	2	2
1	:	1	:	1	1	:	:
:	:	2	:	:	3	1	3
3	1	1	:	1	:	:	2
1	1	:	1	:	:	2	3
2	3	2	:	:	1	2	3
3	1	2	3	2	3	2	:
:	1	2	1	3	1	:	2
3	3	:	1	2	:	:	3
3	1	1	1	2	2	1	1
1	3	2	2	3	1	1	1
1	3	2	1	1	:	3	:
1	2	1	1	:	3	2	1
:	:	1	2	3	3	3	2
:	3	3	3	:	3	2	2
3	2	3	1	1	:	:	2

Figure 3.

shows the recovered plaintext (the recovered background is not shown in the figure).

### 3.2 Two-states picture ( $k = 2$ )

Consider an example in which there are only two states,  $x_{ij} = 0, 1$  (e.g. two colours of a picture). The data are contained in an  $8 \times 8$  matrix ( $c_{ij}$ ) in which the matrix elements along three major diagonals are equal to unity (first pattern of Fig. 2), all other matrix elements being zero. The background matrix ( $b_{ij}$ ) is identically zero and is not shown in the figure. The key is chosen as the asymmetric totality function which is a sum over all nearest neighbouring matrix elements of  $x_{ij}$  except  $x_{i,j-1}$ :

$$f[\{x_{ij}\}] = x_{i-1,j-1} + x_{i+1,j-1} + x_{i-1,j} + x_{i+1,j} + x_{i-1,j+1} + x_{i,j+1} + x_{i+1,j+1}. \quad (7)$$

The results of four iterations are shown in Fig. 2.

### 3.3 Four-states picture ( $k = 4$ )

In this case four colours are considered, i.e.  $x_{ij} = 0, 1, 2, 3$ , with each number representing a colour. Again the asymmetric totality function equation (7) is used. The results are shown in Fig. 3.

## 4. DISCUSSION

As can be seen from the above examples, the encryption described is a very efficient operation; the original

pattern (plaintext) is almost totally 'diffused' in a couple of iterations. Note that the background is arbitrary and can be considered as information to be transmitted; one can transmit two plaintexts at a time. Or if one wants, one can cut a message into two equal halves, one half of it being used as the 'background' and the other half as 'data'; upon decryption, one can recover both the 'background' and the 'data'. It is thus obvious that the ciphertext can always be made just as long as the original text. For a black-and-white picture, each cell (pixel) can have two different states, namely, black (0) and white (1). If the key  $f[\{x_{ij}\}]$  is a function of  $x_{ij}$  and its eight nearest neighbours, the total number of distinct functions is  $2 \wedge (2^9)$  (approximately equal to  $10^{155}$ ), which is much larger than the age of the universe in any physical time unit; it is impossible to obtain the key by trial-and-error. If the picture has colours (i.e. each cell has more than two states), or more neighbours have been considered in the arguments of  $f[\{x_{ij}\}]$ , the number of choices of  $f$  will be much larger. Of course, among those distinct functions, a few of them such as  $f = 0$  or  $f = 1$  cannot be used in practice; also, if  $f$  has symmetry about  $i$  and  $j$ , the ciphertext may preserve certain symmetric pattern from the plaintext, or the iteration may have periodic properties and thus one may avoid using this kind of key. However, the choices of  $f$  are still large enough that none can guess it correctly. For the case of one-dimensional messages, each cell may contain one of the 256 generalised ASCII characters (i.e. 256 states). If  $x_i$  and its eight neighbours are utilised in the construction of the key, the number of distinct functions is  $256 \wedge (256 \wedge 9)$ . This number of choices of  $f$  is truly an astronomical number.

Prior to our model, encryption methods such as Caesar cipher, Vigenere cipher or the Hill cipher are basically alphabetic substitution or matrix transposition (for details, please refer to Refs 1–5). In these models one may obtain the relation between certain characters or the order of characters of a plaintext and the ciphertext by symbol frequency analysis or by knowing part of the text. Once such relations are found, the code may be broken as the relations apply for other sets of plaintext and ciphertext. In our model, there is no fixed relation between such characters (unless one has chosen a very special key like  $f[\{x_i\}] = x_{i+j}$  with a special background). After the first iteration, the state (or character) at a cell may not (or may) depend on itself; it may depend on its nearest neighbour on the right or its second nearest neighbour on the top or both; or it may depend on a certain element at the far left end corner or an element at the centre of the plaintext; or it may depend heavily on a right-side neighbour and depend slightly on a left-side neighbour. The exact dependence is governed by our choice of the function key (of course the actual ciphertext also depends on the background, which is arbitrary). If the function key involves more than one matrix element, after a certain number of iterations, the information at various positions will be mixed up as a consequence of diffusion; the information at the centre may diffuse into the boundary and the information at the boundary may diffuse to the centre. The state at each cell has a little bit of information of each of the other cells of the original plaintext. This is in analogue with the case where we shed many coloured water droplets on to a bathtub full of water; the coloured droplets will diffuse out. After a while they will be mixed up and by just looking at the

colour of the final mixture no one can tell what the original distribution of the coloured droplets was. In this analogy, the process of diffusion is governed by physical laws.

One may ask, if the colour of the original picture at a region has diffused into another region and got mixed up there, how can we go ‘backward’? Is there any guarantee that the original picture can be recovered? The answer is yes. The diffusion of the colour of the picture is governed by a second-order equation which has time-reversal symmetry. If we had used an equation which was not second-order (for example first-order or third-order), there would have been no guarantee that the process could be reversed. In the analogue of coloured water droplets, if the velocities of all the molecules could be reversed, the original distribution of the coloured droplets would be recovered. This can be done by video-taping the process and playing the tape backward. This result is well known in the field of physics and is discussed in almost every textbook of classical mechanics. But it does not mean that it can be derived easily; this well-known result is the product of the endeavour of many people in the nineteenth century.

One way to attack a traditional encryption code is by guessing the content of the first message. For example, the first plaintext may consist of 'PLEASE LOGIN: USER ID: PASSWORD:'. In this model, even if the first plaintext is exactly known, the key function cannot be obtained because of the arbitrary background; if each plaintext consists of 20 characters (the one-dimensional case), there are  $(256)^{20}$  possible backgrounds which is also an astronomical number. (Note again that the receiver does not need to have any knowledge about the background in order to decrypt the ciphertext.)

Another property of this method is that the decrypting operation uses a key-function which is identical to that used in the encryption. In other words, the encrypting device used by the sender is identical to the decrypting device used by the destined receiver. Thus one only has to design one machine which can be used for both encryption and decryption; this makes the data flow bidirectional.

In closing, we would like to point out that there should be an invariant associated with equation (1) as it has time-reversal symmetry. Despite its existence, it is a rather formidable task to derive a general form for the invariant. We have only obtained the solution of the invariant 1 for the very special case

$$x(T+1) = Ax(T) - x(T-1), \quad (8)$$

where  $A$  is an arbitrary constant. The solution for the invariant of equation (8) is

$$I = [x(T+1)]^2 + [x(T)]^2 - Ax(T+1)x(T). \quad (9)$$

## 5. CONCLUSIONS

We have presented a data encryption model based upon time reversal transformations in analogy to a mechanical system whose dynamics is described by second-order (in

## REFERENCES

1. A. S. Tanenbaum, *Computer Networks*, chapter 9. Prentice-Hall, N.J. (1981).

**Table 1. All possible functions of two two-state elements; the equivalent Boolean functions are shown on the right-hand column with symbol V representing OR, \* representing AND;  $\bar{X}$  is the complement of X.**

$X_{i+1}$	: 1 1 0 0	Equivalent boolean function
$X_i$	: 1 0 1 0	
$f[X_i]$	: 0 0 0 0	$f = 0$
	0 0 0 1	$\bar{X}_i * \bar{X}_{i+1}$
	0 0 1 0	$X_i * \bar{X}_{i+1}$
	0 0 1 1	$\bar{X}_{i+1}$
	0 1 0 0	$\bar{X}_i * X_{i+1}$
	0 1 0 1	$\bar{X}_i$
	0 1 1 0	$\bar{X}_i * X_{i+1} \vee X_i * \bar{X}_{i+1}$
	0 1 1 1	$\bar{X}_i \vee \bar{X}_{i+1}$
	1 0 0 0	$X_i * X_{i+1}$
	1 0 0 1	$\bar{X}_i * \bar{X}_{i+1} \vee X_i * X_{i+1}$
	1 0 1 0	$X_i$
	1 0 1 1	$X_i \vee \bar{X}_{i+1}$
	1 1 0 0	$X_{i+1}$
	1 1 0 1	$\bar{X}_i \vee X_{i+1}$
	1 1 1 0	$\bar{X}_i \vee X_{i+1}$
	1 1 1 1	1

time) equations. Principles and properties of the model have been described. We have also presented three distinct examples to highlight the model; they demonstrate that the model is an efficient and useful method to encrypt and decrypt confidential data. The employing of physical concepts in this method may shed light on the application of physical principles to higher-level computation systems.<sup>9</sup>

## APPENDIX

In this appendix we explain in more detail our statement in the text that for a matrix element  $x_{ij}$  there exists  $k \wedge (k \wedge L)$  distinct functions, if  $L$  matrix elements (each can have  $k$  different values) are allowed to be utilised in the construction of the function. A function (or mapping) is simply a rule which assigns to each member of a domain a uniquely determined corresponding member in the range. Therefore the number of distinct functions considered here is simply the number of different ways that we can map the  $L$  matrix elements to  $k$  different states. Since each of the  $L$  matrix elements can take on  $k$  different values, the number of different ways to tabulate their values is  $k^L$ . Therefore the domain of the mapping has  $M = k^L$  different members and the range has  $k$  different members (states). How many ways can we map  $M$  different members to  $k$  different members? Obviously the answer is  $k^M$ . For example, the number of different ways to map  $(a, b, c)$  to  $(0, 1)$  is 8. Thus there exist  $k \wedge (k \wedge L)$  different mappings, if  $L$  matrix elements are utilised in the construction of the function. To further highlight this point, let us consider the simplest case,  $k = 2$ ,  $L = 2$ ; the functions are simply the well-known Boolean functions. According to what we have just discussed, the number of different functions is  $2 \wedge (2 \wedge 2) = 16$ . All these 16 different functions are depicted clearly on Table 1.

3. D. W. Davis and W. L. Price, *Security for Computer Networks*. Wiley, New York (1984).
4. D. E. Denning and P. J. Denning, *Data Security*. *ACM Computing Surveys* **11** (3) 227-249 (1979).
5. D. Kahn, *The Codebreakers*. Macmillan, New York (1967).
6. J. D. Jackson, *Classical Electrodynamics*, 2nd edn. Wiley, New York (1975).
7. J. von Neumann. In *Automata Studies*, edited C. E. Shannon and J. McCarthy, *Annals of Mathematics Studies*, Vol. 34. Princeton University Press, Princeton, N.J. (1956).
8. R. G. Stein, *Fundamentals of College Algebra with Trigonometry*. Nelson-Hall, Chicago (1986).
9. C. Mead and L. Conway, *Introduction to VLSI Systems*, chapter 9. Addison-Wesley, New York (1980).