Cryptography based on Chaotic Synchronization: Round III

P G Vaidya and Sajini Anand

National Institute of Advanced Studies, Indian Institute of Science Campus,
Bangalore 560 012, Email: pgvaidya@nias.iisc.erent.in

Abstract. This paper discusses cryptography based on the property of chaotic synchronization. Specifically, it is about Round III of such a cryptographic method. Round I showed the feasibility of using chaotic synchronization for cryptography. Round II consisted of a method to counter attack. This paper is Round III and shows how to counter the counter attacks. First, we show numerical evidence that synchronization is possible between two Lorenz systems if one system sends information about x_0 at a slower rate. The second system evolves on its own, except that when it receives a signal from the first system, it replaces its own value of y_0 by the received x_0. We have found that the two systems eventually synchronize, but often after a long time. Therefore, we have devised a technique to speed-up this synchronization. Once this is done, it is possible for the authorized receiver (with the possession of the initial super-key) to keep synchronizing with slowly sampled inputs, whereas the known methods of Round II do not help an eavesdropper.

Keywords: Chaotic Cryptography, Synchronization, Secure Communication, Super-key.

1 Introduction

One of the most well known characteristics of chaotic system is its sensitivity to initial conditions. It was therefore a surprise to see the result, first reported by Pecora and Caroll [1] that chaotic systems can be synchronized sending only a part of the state space information from one system to another. Soon it was suggested [2, 3] that this can be used to create keys for cryptography using the unsent state spaces (Round I). This system was first used in connection with the Lorenz equation. The assumption was that access to the sent information is impossible without knowledge of the three parameters of the equation. This is why the knowledge of the three parameters is known as the super-key. Exhaustive search for this key from the existing data is time consuming and can easily be countered by changing the key frequently.

However, in Round II, it was shown that this cryptosystem is easily breakable and the super-key can be found in a very rapid manner from the synchronizing signal. This was possible if the synchronizing sampling rate. In that case, a new

method to calculate the derivatives of the variables was used to arrive at a non-linear equation for the super-key. This problem was then embedded in a higher dimension to get a linear equation. The solution rapidly gave the super-key.

For Round III, we wanted to synchronize the system by sending signals at such large intervals that the methods of Round II would fail.

However, we found that if the synchronizing signals are not very frequent, the systems synchronize but it takes a very long time. Therefore, our main result is about how to force synchronization quite rapidly. Once this is done, the sender and receiver can frequently communicate and change the parameters of the equation so that the task of the eavesdropper becomes even more difficult.

In what follows, first we would briefly review these developments of Round I and II. Then, we would describe the new results which makes it possible to counter the results of Round III.

2 Round I: Feasibility of using Chaotic Synchronization for Secure Communication

We will assume that we have a sender ("Alice") who generates some data using a differential equation. As a specific example, we will consider the Lorenz equation:

$$\frac{dx_0}{dt} = \sigma(x_1 - x_0) \tag{1.1}$$

$$\frac{dx_1}{dt} = \rho x_0 - x_1 - x_0 x_2 \tag{1.2}$$

$$\frac{dx_1}{dt} = x_0 x_1 - \beta x_2 \tag{1.3}$$

The parameters of this equation are α, ρ and β. These three constitute what is collectively know as the *Super-key*. We assume that the authorized receiver ("Bob") has received this super-key with an alternative method (typically a public-key method such as RSA). We will assume that an eavesdropper does not know the super-key.

Now, let us represent Bob's system by

$$\frac{dy_0}{dt} = \sigma(y_1 - y_0) \tag{2.1}$$

$$\frac{dy_1}{dt} = \rho y_0 - y_1 - y_0 y_2 \tag{2.1}$$

$$\frac{dy_2}{dt} = y_0 y_1 - \beta y_2 \tag{2.1}$$

In the original system by Pecora and Caroll [1] Bob's y_0 was completely over-ridden by Alice's x_0. So Bob really had only two equations

$$\frac{dy_1}{dt} = \rho x_0 - y_1 + x_0 y_2 \tag{3.1}$$

$$\frac{dy_1}{dt} = x_o y_1 - \beta y_2. \tag{3.2}$$

In the system in Round III, we will continue to use the full equations for Bob, except that we have an option to override his y_0 by x_0 whenever information about x_0 is available.

Coming back to Round I, it was found that after a passage of time (which was later shortened [3])

$$y_1(t) = x_1(t)$$

and

$$y_2(t) = x_2(t).$$

This is known as synchronization.

This was at first a surprise because the systems are chaotic. In [2], this property was proven by using Lyapunov function and it was also suggested for the first time that this can be used for cryptography (Fig. 1).

The suggestion was to use the unsent $x_1(t)$ and $x_2(t)$ to create keys. These methods were further illustrated in (for example, see [4,5,6]).

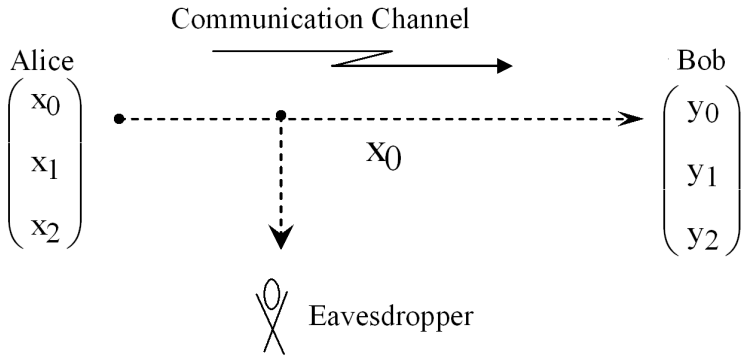

Fig. 1. Alice sends $x_0(t)$ on an open channel to Bob. Eventually Bob's system will synchronize with Alice, i.e. $x_1(t) = y_1(t)$ and $x_2(t) = y_2(t)$

Alice also uses $x_1(t)$ and/or $x_2(t)$ to generate keys. These keys are used to encrypt a message which is also sent on the open channel. Bob, synchronizes with Alice and then uses $y_1(t)$ and $y_2(t)$ to get the key sequences and decrypts the message.

3 Round II: Breaking of the Super-key

In [7] it was shown that an eavesdropper can determine the super-key from the knowledge of $x_0(t)$ alone.

This can be done with a very small segment of $x_0(t)$. This was important because Alice and Bob, once synchronized, can rapidly communicate changes in the super-key.

The breakthrough was achieved by first transforming the Lorenz equation to a canonical form. Then they used a new method to find highly accurate derivatives from data. Next, they transformed a nonlinear equation for the super-key to a linear form by embedding it in four dimensions. The final equations were solved by using the generalized inverse. This method used the fact that $x_0(t)$ is sent to Bob in a virtually continuous manner (in fact, the theory assumed a continuous signal). In any case, the samplings were quite dense. It was mentioned in their paper that if the samples are not dense the method fails.

At that stage, this was not a serious objection. Because, if the samples are not frequent, the systems take a long time to synchronize and Alice has no idea whether Bob has synchronized. Even the method of [3], which speeded up this procedure, assumed that the sampling was quite fast.

4 Round III: Development of a System which is Immune to Round II

Round III depends on sending only intermittent samples of $x_0(t)$ on the open channel. First, we would show that in this case, synchronization is still possible. Then, we would show how to speed this process. To see how synchronization is possible, we generated Alice's equations $x_0(t), x_1(t), x_2(t)$ using equations (1).

Parameters of the system are ($\sigma = 10.0, \beta = 8/3, \rho = 29.75$) and the initial conditions are $\begin{pmatrix} 1.874 \\ 2.056 \\ 19.142 \end{pmatrix}$

Bob does not know these initial conditions, except for x_0 which is received.

He uses the full equations (2). When he receives an information from Alice, he resets y_0 to the value of x_0 received.

In the first case shown in Figs 2 and 3. Alice's trajectory was calculated (using the Runge-Kutta procedure), at every 0.0001 second. However, information was sent to Bob every 0.1 second. It can be seen from Figure 3 that it takes at least 5 seconds for the two to synchronize. If his initial guess is different, it might take less time or it might take longer. Alice has no way to know when this has taken place. If the information is sent at a longer interval, it takes even longer time to synchronize.

5 Rapid Synchronization

In this section, we would show how synchronization is possible with only first three samples from Alice.

To get the overall idea behind this, consider Fig. 4. In this concept, we conceive of a map from initial conditions in the usual state space of x_0, x_2, x_2 and

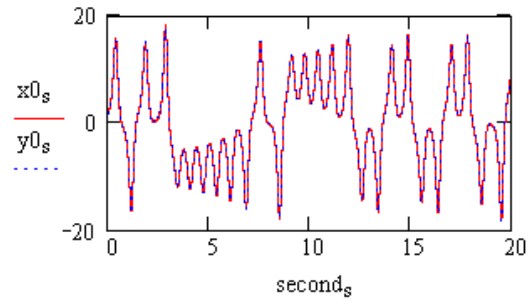

Fig. 2. Alices signal Vs Bob signal as a function of time. Bob's trajectory eventually synchronizes with Alice after along time

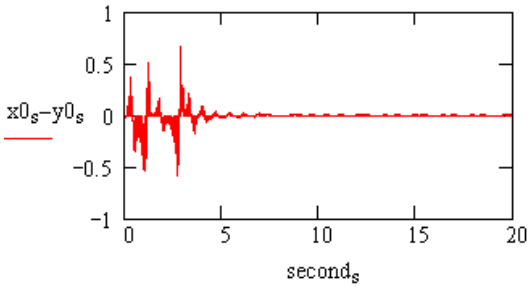

Fig. 3. The difference of Bob and Alices trajectories plotted against time. Bob's trajectory eventually synchronizes with Alices

a new state space constructed from the samples of x_0. Here, (for the case of samples being sent every 0.1 seconds).

$$S_0 = x_0(0)$$
$$S_1 = x_0(0.1)$$
$$S_2 = x_0(0.2).$$

The main idea is that if we choose some other initial conditions in (x_0, x_1, x_2) space, we would get another point in the S_0, S_1, S_2 space. In fact, in all probabilities, Bob chooses another. At that point he can be sure that his x_0 (and therefore S_0) would agree with Alice's but x_1, x_2 (or as we have used the notation y_1, y_2) would be different.

The main idea is to look at the S picture and find how Alice and Bob find a correction for Bob's initial condition.

Once again the approach of trial and error runs into enormous difficulties. The method which works here can be seen as a generalization of Newton-Raphson method of higher dimension.

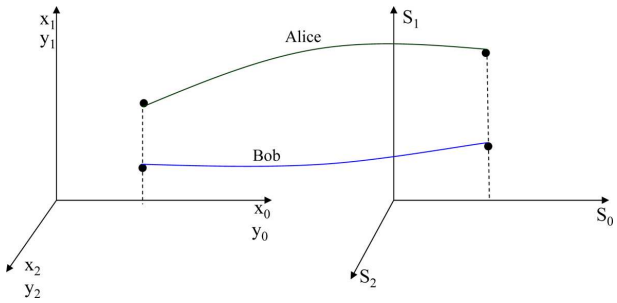

Fig. 4. A sketch of a map between initial conditions and measurements x_0 at $t = 0$, 0.1 and 0.2. Bob's guess results in discrepancy in S space

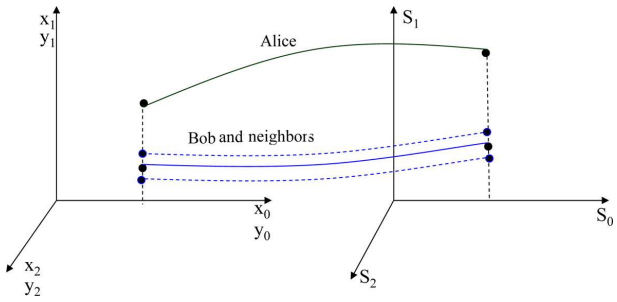

Fig. 5. Map of Alice's, Bob's and his neighbors initial conditions and observations

So essentially Bob starts with some initial guess and generates some trajectories quite nearby him. Based upon these trajectories what he tries to do is to develop a relationship between deviations from his own trajectory in y_1 and y_2 dimension and how they relate to the deviations in the S components at $t = 0.1$ and 0.2 secs respectively. We can see that the errors in the initial trajectories lead to discrepancies in the observations of the component later. If the errors are small, then the discrepancies later would be linearly related to these errors and can be represented by a matrix. This matrix can be inverted; therefore from errors we can guess back what the initial discrepancies were. Using the same procedure, Bob doesn't know where Alice has started in terms of x_1 and x_2. But Bob does know what discrepancies occur with the Alices signals in his own trajectories. Using those discrepancies and the same inversion matrix, Bob can improve his case. If Bob's origin is fairly close to Alice's, then the case is improved almost immediately to the exact case. But if Bob is farther, several iterations would be needed. It has been found that when every 1000 sample is taken, convergence took place in every case no matter where Bob started his trajectory. But in the case of every 5000 samples such convergence require that

the region needed to be split in to different regions and then find out which among them leads to convergence.

Specially, let us take two additional trajectories which in principle start very close to Bob's initial conditions. He knows where he started and where the neighbors started. Bob should know the deviation from the distance between him and his neighbors, which leads to the deviations in his y_0 component and his neighbors and hence the relation between those deviations and the initial conditions. We could do this in principle by taking extremely small deviations from Bob's starting point, using exact equations and finding the transfer matrix. In practice this can be accomplished much better by taking the Jacobian of the non linear equation over a very small region. This scheme is shown in Figure 5.

Returning to Fig. 4, Bob does not know exactly where Alice starts in x_0, x_1, x_2 space. He does choose $y_0 = x_0$ and some y_1 and y_2. Now, he runs a simulation using equation (2) and arrives at a point in the S space. In the S space at $t = 0$, the values of S_0 for both are identical. Let us assume that at the next observation at $t = h$ (e.g. $h = 0.1$ sec) Bob's y_0 is $(S_1 - \delta)$ and at the next one at $(t = 2h)$ it is $(S_2 - \mu)$. We would form a column vector using this derivation, called Jump. So,

$$Jump = \begin{bmatrix} S_1 - (y_0)_h \\ S_2 - (y_0)_{2h} \end{bmatrix} = \begin{bmatrix} \delta \\ \mu \end{bmatrix}. \tag{4}$$

Consider one of the neighbors of Bob. Assume that its trajectory is given by

$$\begin{pmatrix} y_0(t) + \eta_0(t) \\ y_1(t) + \eta_1(t) \\ y_2(t) + \eta_2(t) \end{pmatrix}.$$

If η's are very small, the equation for them can be found from the Jacobian of Bob's equation. Thus

$$\frac{d}{dt}\begin{pmatrix} \eta_0 \\ \eta_1 \\ \eta_2 \end{pmatrix} = \begin{bmatrix} -\sigma & \sigma & 0 \\ (\rho - y_2) & -1 & -y_0 \\ y_1 & y_0 & -\beta \end{bmatrix} \begin{pmatrix} \eta_0 \\ \eta_1 \\ \eta_2 \end{pmatrix}. \tag{5}$$

We can solve 3 sets of equations simultaneously. One for Bob and two others which both follow equation (5). For both the neighbors η_0 is zero. Since we can extend the linearization, we can choose $\eta_1 = 1, \eta_2 = 0$ for one neighbor and $\eta_2 = 1$ for the other.

Now, define matrix A as

$$A = \begin{bmatrix} (\eta_0)_h & (\eta_0')_h \\ (\eta_0)_{2h} & (\eta_0')_{2h} \end{bmatrix} \tag{6}$$

where the prime is used for the second neighbor.

The A matrix tells us how unit derivations in x_1 and x_2 initial conditions transform into derivations in S_1 and S_2.

If linearity were to prevail the initial error in Bob's position:

$$Error = \begin{pmatrix} y_1(0) - x_1(0) \\ y_2(0) - x_2(0) \end{pmatrix} \tag{7}$$

will also get multiplied by the same matrix so that

$$Jump = A \cdot Error. \tag{8}$$

Therefore,

$$Error = A^{-1} \cdot Jumb \tag{9}$$

$$Corrected\ y = \begin{bmatrix} y_0 \\ y_1 + Error_0 \\ y_2 + Error_1 \end{bmatrix}. \tag{10}$$

In general, linearity will not extend to Alice's position. So, following the spirit of Newton–Raphson method, he chooses the corrected value and iterate again. If this procedure does not converge, he selects an initial condition from a different part of the state space.

6 Numerical Results

We continue with equation (1) for Alice. We use Runge-Kutta procedure and find x at intervals of 10^{-4} sec. Initial conditions were chosen at

$$\begin{pmatrix} 1.874 \\ 2.056 \\ 19.142 \end{pmatrix}.$$

In the first simulation, we used $h = 0.1$ sec. Therefore

$$S_0 = x_0(0); \quad S_1 = x_0(0.1); \quad S_2 = x_0(0.2).$$

Bob knows S_0, S_1 and S_2.

We choose several initial guesses for Bob. One which was quite for away from Alice was

$$\begin{pmatrix} 18.46 \\ 0 \\ 0 \end{pmatrix}.$$

(Of course, $y_0 = x_0 = S_0$, because Bob knows it.)

The iteration procedure worked quite well. Bob's predicted trajectory is shown in Fig. 6. The derivations shows in Fig. 7 shows that the synchronization is almost perfect.

Our second simulation used the same initial conditions for Alice but h was 0.5 seconds, so that

$$S_0 = x_0(0); \quad S_1 = x_0(0.5); \quad S_1 = x_0(1.0).$$

Now if Bob starts with the guess

$$\begin{pmatrix} 18.46 \\ 0 \\ 0 \end{pmatrix}.$$

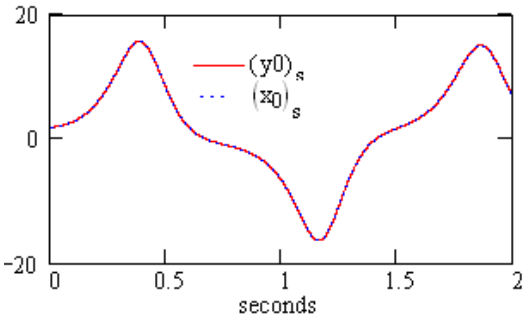

Fig. 6. Evolution of x_0 and predicted y_0 for $h = 0.1$ seconds

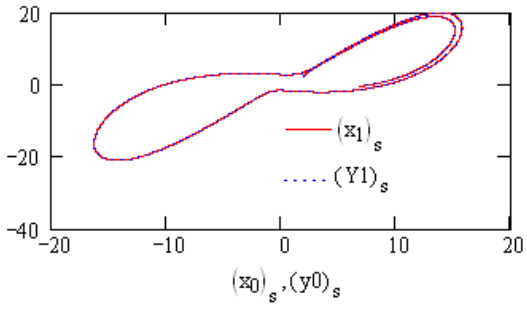

Fig. 7. State space for x and y

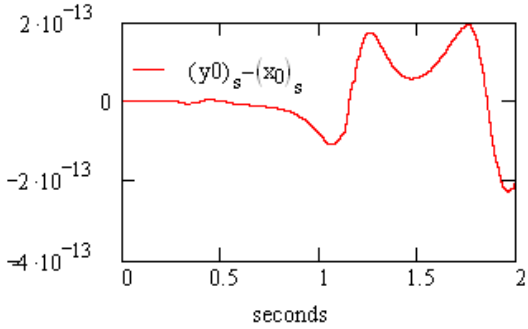

Fig. 8. Difference in x_0 and y_0

it does not converge. However, there is a fairly large neighborhood of initial conditions around Alice's conditions for which convergence takes place. So, using

a strategy akin to simulated annealing, we soon arrive at an initial condition

$$\begin{pmatrix} 18.46 \\ 1 \\ 16.5 \end{pmatrix}.$$

In this case, the synchronization is once again quite good. This is shown in Figs 9, 10 and 11.

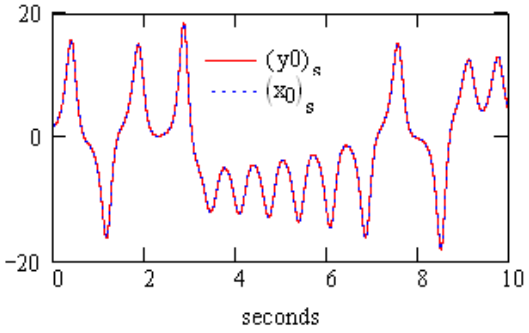

Fig. 9. State space for $h = 0.5$ second

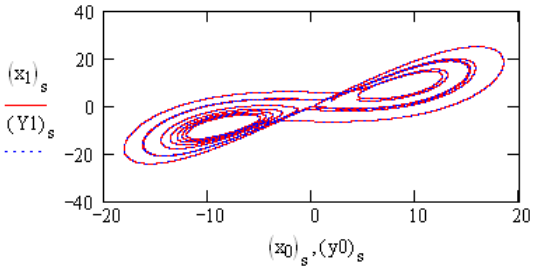

Fig. 10.

7 Conclusions

The theory of chaotic synchronization assumes a continuous feedback from sender to receiver. In practice we send digitally sampled versions, with a fairly high sampling rate. But an eavesdropper can find the super-key in such a case as shown in

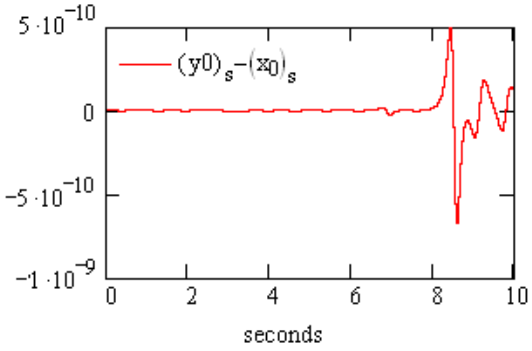

Fig. 11.

[3]. So we counter this possibility by infrequent sampling. Here receiver achieves rapid synchronization and once the systems are synchronized the super-keys can be changed frequently. Hence eavesdropper cannot attack the private communication, at least by the published method.

References

1. L. M. Pecora and T. L. Carroll, *Phys. Rev. Lett.* **64**, 821 (1990)
2. Rong He and P. G. Vaidya, Analysis and synthesis of synchronous periodic and chaotic systems, *Phys. Rev.* **A46**, 7387–7392 (1992)
3. P. G. Vaidya, Monitoring and speeding up chaotic synchronization Chaos, Solitons and Fractals, Volume 17, Number 2, July 2003, pp. 433-439(7)
4. Rong He and P. G. Vaidya, Implementation of chaotic cryptography with chaotic synchronization, *Phys. Rev.* **E57**, 1532-1535 (1998)
5. K. M. Cuomo and A. V. Oppenheim, *Phys. Rev. Lett.* **71**, 65 (1993
6. M. Lakshmanan and S. Rajasekar, Nonlinear Dynamics: Integrability, Chaos, and Patterns, Springer-Verlag, 2003, Ch. 16, India
7. P.G. Vaidya and Savita Angadi, Decoding chaotic cryptography without access to the super key. *Chaos, Solitons and Fractals* **17(2-3)**, 379-386, (2003)