Abstract—Cryptography is the science of hiding important information while transmitting over an insecure channel making it impossible for any adversary to read.Cryptography is very important for transmission and sharing of confidential information preventing any misuse of it.Neural Networks is a mathematical model which simulates the structure and functionality of biological neural network.A chaotic neural network is a network which adds randomness to a signal which is extremely hard to predict.Adding a Chaotic Neural Network to a Cryptographic system enhances the security of the system making it difficult to decode by the adversaries.In this research paper we have collaborated a Chaotic Neural Network with an Elliptic Curve Cryptographic System which is then compared with the conventional models such as RSA,Blowfish and RC2 models and is found to be better than various models based on certain parameters.

1.1. Concept of Artificial Neural Network

Artificial Neural Networks are mathematical models which are used to imitate a biological neural network. ANN’s are used to approximate or estimate functions which require a lot of inputs. These mathematical models are used where the human brain is considered to be more efficient in computing a typical function as compared to a machine e.g identifying various images,or identifying different handwritings etc.

Neural Network is a network of nodes known as neurons which are connected by directed links, where every link \((X,Y)\) that connects the node X and Y has a weight \(w\), and there is an activation function \(f\) which depending upon a threshold value maps the input state into an output state[24]. Neural Networks can be differentiated into two types: Feedforward Networks and Recurrent Networks. A Feedforward Neural Network is acyclic and thus it does not have any other state rather than the weights themselves. On the opposite side, a Recurrent Network is a cyclic network where the output is fed back into the network along with weights. Because of this feedback, Recurrent Networks need small amount of memory to store that data.
1.2. Related Work

In 2004, Einat Klein presented cryptography framework which was based on a secret-key generation by a neural network in a public channel [4]. The model he proposed had two neural networks trained on their alternate outputs syn-chronized to an equal time dependent weight vector through a chaos synchronization system starting with different initial conditions. In 2011, R. M. Jogdand proposed a model where secret key was generated by neural networks [8]. The neural cryptography has two communication networks which receive an identical input vector, generate an output vector and are trained on the output vector. Also in 2012, Pratap Singh generated a secret key over a public channel using a neural network [9]. The model consists of 2 partners which had an input vector and different initial conditions which were synchronised by a common external signal and received a common input vector with the machine predicting their output by mutual learning. Zhang, proposed a novel chaotic keyed hash function using a feedforward feedback non-linear filter. Arumugam studied the effective use of logistic map and Lorenz map in generating message authentication codes [25]. In 2014, Xiang T. proposed a method where the image is compressed and encrypted by a chaotic map and arithmetic encoding, the encryption and compression were done at different stages making it possible for the adversaries to break the cryptosystem without getting through the compression stage. In 2013, H. Zhu and C. Zhao proposed a new encryption and compression scheme using a hyper-chaos and Chinese Remainder Theorem [3]. It yielded a correlation coefficient of 0.0058 and an entropy of 7.98 but its drawback was that it could be easily broken by the adversaries due to its similar plaintext encryption nature.

2. Preliminaries

2.1. Simulation of a Chaotic Neural Network

The Chaotic Neural Network used in our research work is based on 2 frequency mods which is a modified version of Hebb’s Law. The equations for this model are described in this way:

\[ x_i(t + 1) = (1 - D)x_i + E_g(x_i) \]

where

\[ g(x_i(t)) = \sum_{j=1}^{N} w_{ij} f_j(x_j(t)) + I_i(t) \]

Here D is the decay parameter of potentials, E is the excitatory rate, I(0) is the external input of the i'th neuron.

2.2. Elliptic Curve Cryptography

The locus of a point, whose coordinates conform to a particular cubic equation along with the point at infinity \( \infty \) (the point at which the locus in the projective plane intersects the line at infinity) is known as an elliptic curve.

The equation of \( E(F_p) \) for the characteristic \( p \) > 3 can be defined as

\[ y^2 = x^3 + ax + b \] (2)

where \( a, b \in F_p \) and \( b \neq 0 \).

Key Generation

Key Generation for an Elliptic Curve Cryptosystem is as follows:

Every node needs to have a pair of public and private keys. Every sender will be Encrypting the message with the receiver's Public Key and the receiver will Decrypt the message using it's private key.

Step 1: Select a number "d" in the range of "N" where N is a prime number.

Step 2: Now we can generate the Public Key "Q" as follows:

\[ Q = d \cdot P \] (3)

where P is a point on the Curve

Step 3: d is our Private Key and Q is the required Public Key.

Encryption

Two Ciphertexts must be generated and let the ciphertexts be C1 and C2. They are generated as follows:

Step 1: Choose a number "K" less than N.

\[ C1 = k \cdot P \] (4)

\[ C2 = M + k \cdot Q \] (5)

where M is the Message we have to send

Step 2: Now send both of the Ciphertexts.

Decryption

We can get the original message back as follows:

\[ M = C2 \cdot d \cdot C1 \] (6)
3. Methodology

Encryption and Decryption Processes

Figure 2. Encryption Using a Chaotic Neural Network

\[
M(n)(1 - x(n))
\]

STEP 4: Create \(b(0), b(1), ..., b(8M - 1)\) from \(x(1), x(2), \ldots, x(M)\) by the generating scheme that \(0.b(8m-8)b(8m-7) \ldots b(8m-2)b(8m-1) \ldots\) is the binary representation of \(x(m)\) for \(m = 1, 2, \ldots, M\).  

STEP 5: The weight \(t\) and biases \(O\) for all the neurons in the hidden layer are calculated given below.

\[
\text{For } n : 0 \text{ to } M-1 \text{ Do :}
\]

\[
g(n) = \sum_{i=0}^{7} d_i 2^i \quad \text{(7)}
\]

\[
X_i
\]

For \(i=0\) to \(7 \text{ Do:} \)

if \((j \text{ equals } i \text{ and } b(8*n + i) \text{ equals } 0) \)

\(t=1\)

else if \((j \text{ equals } i \text{ and } b(8*n + i) \text{ equals } 1) \)

\(t=-1\)

else if \((j \text{ not equals } i) \)

\(t=0\)

\[
g'(n) = \sum_{i=0}^{7} d_i 2^i \quad \text{(8)}
\]

Now we can get the final transformed data in bits using the following algorithm.

\[
g'(n) = d_i 2^i \quad \text{(9)}
\]

\(g'(n)\) is the required data which is to be encrypted using Elliptic Curve Cryptosystem and sent across.

The next part is to generate Private and Public Keys for the CryptoSystem.

As described above, The Elliptic Curve Cryptosystem requires to generate a Private Key which is chosen randomly by the Computer. The Public Key of every Node is also dependent on this Private Key. Thus to make system more secure, the randomly generated Private Key is again passed through the Chaotic Neural Network to make the keys even more secure. Then the Encryption process is carried out using the modified Public Key of the receiver and sent across the insecure channel.

3.1. Encryption Process on the Sender Side

First the data is divided into blocks of 8 bits each which will be Encrypted using our Algorithm. The blocks which are left are padded with zeroes. Then the chaos is created using the following algorithm by the Chaotic Neural Network.

STEP 1: Set the value of parameter \(M\).

STEP 2: Determine the parameter, \(U\) and the initial point \(x(0)\) of the 1-D logistic map.

STEP 3: Evolve the chaotic sequence \(x(1), x(2), \ldots, x(M)\) by \(x(n+1) = M(n)(1-x(n))\).

STEP 4: Create \(b(0), b(1), \ldots, b(8M-1)\) from \(x(1), x(2), \ldots, x(M)\) by the generating scheme that \(0.b(8m-8)b(8m-7) \ldots b(8m-2)b(8m-1) \ldots\) is the binary representation of \(x(m)\) for \(m = 1, 2, \ldots, M\).

STEP 5: The weight \(t\) and biases \(O\) for all the neurons in the hidden layer are calculated given below.

\[
\text{For } n : 0 \text{ to } M-1 \text{ Do :}
\]

\[
g(n) = \sum_{i=0}^{7} d_i 2^i \quad \text{(7)}
\]

\[
X_i
\]

For \(i=0\) to \(7 \text{ Do:} \)

if \((j \text{ equals } i \text{ and } b(8*n + i) \text{ equals } 0) \)

\(t=1\)

else if \((j \text{ equals } i \text{ and } b(8*n + i) \text{ equals } 1) \)

\(t=-1\)

else if \((j \text{ not equals } i) \)

\(t=0\)

\[
g'(n) = \sum_{i=0}^{7} d_i 2^i \quad \text{(8)}
\]

Now we can get the final transformed data in bits using the following algorithm.

\[
g'(n) = d_i 2^i \quad \text{(9)}
\]
3.2. Decryption on the ReceiverSide

The Decryption on the receive side is exactly opposite in order of the Encryption mechanism. First the Ciphertext is passed into the Elliptic Curve Cryptosystem and decrypted using the Public and the Private keys. The Text decrypted using the Cryptosystem is fed into the Chaotic Neural Network to get the original text.

4. Results

The optimality of the proposed algorithm can be judged based on various parameters. The parameters that we have chosen in our research work are as follows:

Encryption Time Time taken to Encrypt a file of fixed size. The less Encryption time is preferred as the algorithm has to be fast enough to convert the secret message into a hidden message.

Decryption Time Time taken to Decrypt a file of fixed size. The less Decryption time is preferred as the algorithm has to be fast enough to convert the hidden message back into the secret message.

Time to generate keys Time to generate keys should be as less as possible as keys play the most important role in Encrypting and Decrypting data.

Throughput Throughput should be as high as possible as it denotes the amount of data to be transferred from the channel.

Correlation Coefficient Correlation function should be close to 1 as a 0 Correlation Coefficient denotes no change in the secret message and the ciphertext.

The Bar Graph below shows the time required for generating keys. The key size for all the algorithms except our ECC using ANN is 1024 bits and for our algorithm it is 192 bits. This is done because other algorithms with 1024 bits key size provides a same security as ours with 192 bits.

The above graph shows the Decryption time of various algorithms which were used some time ago like Blowfish and some algorithms which are currently in use like RSA. The above diagram shows that for small file RSA outperforms all other algorithms but as the file size increases, other algorithms catch up and with large file size, our algorithm takes the least amount of time in ms.

The above graphs show throughput for various algorithms.
algorithms. Throughput is defined as the number of bytes sent across the channel per unit of time. The Results show that RSA has the highest Throughput with our algorithm comingsecond.

\[
\text{throughput} = \frac{\text{bytessent}}{\text{timeinseconds}}
\]  \hspace{1cm} (10)

Figure 8. Comparison of various algorithms

The above Bar Graph shows the Corelation Coefficient of various algorithms. The correlation coefficient being close to 0 represents that there is not much change in the plaintext and the ciphertext. The Corelation Coefficient for our algorithm is the maximum which shows that our algorithm has changed maximum characters of the plaintext which makes it better than the rest.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RSA</th>
<th>Blowfish</th>
<th>RC 2</th>
<th>ECC. ANN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption time for smallfiles</td>
<td>4.12ms</td>
<td>7.143ms</td>
<td>6.922ms</td>
<td>7.02ms</td>
</tr>
<tr>
<td>Encryption time for largefiles</td>
<td>7.65ms</td>
<td>7.722ms</td>
<td>6.656ms</td>
<td>6.245ms</td>
</tr>
<tr>
<td>Decryption time for smallfiles</td>
<td>4.213ms</td>
<td>6.493ms</td>
<td>5.257ms</td>
<td>5.05ms</td>
</tr>
<tr>
<td>Decryption time for largefiles</td>
<td>5.413ms</td>
<td>6.331ms</td>
<td>5.573ms</td>
<td>5.323ms</td>
</tr>
<tr>
<td>Timetogen keyst</td>
<td>59ms</td>
<td>201ms</td>
<td>29ms</td>
<td>251ms</td>
</tr>
<tr>
<td>Throughput (inMB/sec)</td>
<td>2.35</td>
<td>1.67</td>
<td>1.86</td>
<td>2.01</td>
</tr>
<tr>
<td>Corelation</td>
<td>0.017</td>
<td>0.0015</td>
<td>0.002</td>
<td>0.021</td>
</tr>
</tbody>
</table>

TABLE 1. COMPARISONS OF VARIOUS ALGORITHMS BASED ON DIFFERENT PARAMETERS. 

5. Conclusion
This research work has analyzed on various techniques which are used for transfer of data from one end to another in a safe and a secure manner. Chaotic Neural Network has been used to transfer data from one end to another. This Neural Network do not require any data set as the input is solely dependent on the chaotic sequence, which makes the process almost impossible to crack by the intruder. The Chaotic Neural Network is used to modify an Elliptic Curve Cryptography framework. The model is compared with few other models that are currently in use and is found to be working better than the conventional models in some aspects. Our model of using Elliptic Curve Cryptography using a Chaotic Neural Network has the highest Co-relation Coefficient and second best in terms of throughput after RSA. Also the Encryption and Decryption time is dependent on the size of the file. The bigger files are fastest Encrypted and Decrypted using our model. This type of cryptography can be used at places where secrecy of data is of utmost importance. Chaotic nature makes it very unpredictable for intruder to crack the secretmessage.

References


