Alternate Approach for Implementation of SHA-2 Algorithm using Feed forward Neural Network

Prof.V R Kulkarni
Department of Information Science and Engineering, Gogte Institute of Technology, Belgaum

Dr. S S Apte
Department of Information Technology, Walchand Institute of Technology, Solapur

ABSTRACT

In this paper an algorithm for one-way hash function construction based on a two layer feed forward neural network along with the piece-wise linear (pwl) chaotic map is proposed. Where SHA-2 is cryptographic hash function designed by NSA(National Security Agency).
Based on chaotic neural networks, a SHA-2 Hash function is constructed, which makes use of neural networks' diffusion property and chaos' confusion property. This function encodes the plaintext of arbitrary length into the hash value of fixed length (typically, 128-bit, 256-bit or 512-bit). Theoretical analysis and experimental results show that this hash function is one-way, with high key sensitivity and plaintext sensitivity, and secure against birthday attacks or meet-in-the-middle attacks. These properties make it a suitable choice for data signature or authentication**.**

Keywords

Artificial Neural Network, Hash Function, Feed Forward, Plaintext Sensitivity

1. INTRODUCTION

1.1 One-Way Hash function

Hash function accepts a variable size message M as input and produces a fixed size output, referred to as a hash code H (M).Unlike MAC, a hash code does not use a key but is a function only of the input message. The hash code is also referred to as message digest or hash value. The hash code is function of all bits of the message and provides an error detection capability. A change to any bit or bits in the message results in a change to the hash code. The purpose of a hash function is to produce a "fingerprint" of a file, message or other block of data. Hash function must have the following properties [1].

1. H can be applied to a block of data of any size.

2. H produces a fixed length output.

3. For any given value h, it is computationally infeasible to find x such that H(x) =h. This is sometime referred to in the literature as the one-way property.

4. For any given block x, it is computationally infeasible to find y # x with H(y) =H(x).This is sometimes referred to as weak collision resistance.

5. It is computationally infeasible to find any pair (x, y) such that H(x) =H(y). This is sometime referred to as strong collision resistance [2].

Some of the important hash functions are SHA, MD5 and HMAC etc.

1.2 Artificial Neural network

Artificial neural networks(ANN) are mathematical models of complicated biological neural networks.ANN consists of many computation components in parallel.ANN can also be trained, and together with the potential to process vast amount of data in parallel, can be used for many purposes including pattern recognition and economic prediction [3]. Non-linear functions are used in most of the hash function. Non linear functions make it difficult to obtain the input from the output function. In SHA-2, there are a total of four non-linear functions that takes in three 64 bit input and produces a 64 bit output. The presence of the sigmoid non-linear function along with Piecewise linear Chaotic map suggests that the feed forward network may be suitable for one-way hashing.

2. PROBLEM DEFINITION

A hash function is constructed based on a three-layer neural network. In this paper the applicability of using a feed forward neural network along with piecewise linear chaotic map (non linear function) as a possible hash algorithm is investigated. The difficulty of recovering an input from a feed forward network hashed output is presented. Important features of good hash algorithms such as resistance to birthday attacks or meet-in-the middle attacks and collision free hashing are explored. Additionally, the neural network's property makes it practical to realize in a parallel way. These properties make it a suitable choice for data signature or authentication.

2.1 Motivation

Traditional approach for the implementation of hash function makes use of Boolean algebra, permutation and transposition methods. Different attacks such as statistical attack, birthday attack, and meet-in-the middle attack give the strength to the intruder to analyze these algorithms to find a plaintext whose hash value is same as one of the given plaintexts. Recently, it was reported that such widely used hash functions such as MD5 or SHA-1 are no longer secure.
Neural networks have one-way property. For example, if a neuron has multi-input and single output, then it is easy to obtain the output from the inputs but difficult to recover the inputs from the output [5]. This property makes neural networks suitable for hash function designing. In this paper we use neural networks one way property to construct a one way hash function. In particular, the feed forward network along with the piecewise linear chaotic map (non linear function) will be analyzed regarding its ability to

satisfy the conditions of a good one-way hash function: difficulty of obtaining the input from the output, resistance to birthday attacks and collision resistance [7].

3. NEURAL NETWORK TOPOLOGIES

In the previous section we discussed the properties of the basic processing unit in an artificial neural network. This section focuses on the pattern of connections between the units and the propagation of data. As for this pattern of connections, the main distinction we can make is between: Feed-forward neural networks, where the data flow from input to output units is strictly feed forward. The data processing can extend over multiple (layers of) units, but no feedback connections are present, that is, connections extending from outputs of units to inputs of units in the same layer or previous layers.

The structure of feed forward network is made up of layers of neurons. For the purpose of the one-way hashing function, two layers of neurons are employed, the preceding layer is called the "hidden" layer, and the other called the "output" layer. The graphical structure of a fully connected feed forward network is shown in figure 3.

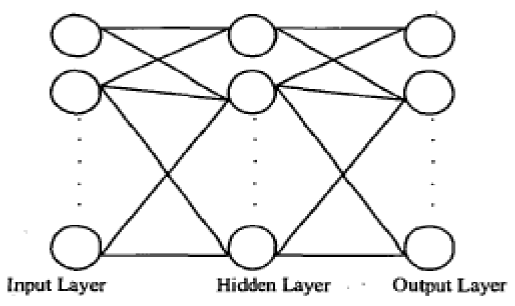


Figure 1: Structure of Feed forward network

Input bits are fed into the "hidden" layer of neurons. The output of the "hidden" layer becomes the input for the "output" layer. Due to the sigmoid function, the output of each neuron is a real number between 0 and 1.Since it will be more useful to have binary values of either 0 or 1, a threshold of 0.5 is taken. Real outputs less than 0.5 will be taken as 0, while those greater or equal to 0.5 will be taken as 1.

The weights Wi are taken from a consistent source. A good source of weight values can be generated from pseudorandom number generators that follow a Gaussian distribution with a mean of 0 and a variance of 1.It is generally advisable to use values that are between -1 and 1 to prevent a neuron from being saturated. The value of the weights in the feed forward network is required to be the same every time the network is initialized for use as a one-way hash function.

The model of the feed forward network used in this project has an output layer consisting of 256 nodes, with a hidden layer consisting of 64 nodes. The number of input bits is set to 768.The total number of weight values including bias weight values, n required is given by the equation

$$\mathbf{n = (256*65) + (64*768) = 65792}$$

3.1 The implemented SHA-2 Hash function

For implementation MATLAB 7.7 version software is used. Hash function accepts a variable size message M as input and produces a fixed size output, referred to as a hash code H (M).

(1) Since the input can be of variable number of bits long, padding is done in a similar fashion performed in SHA-2 .A single '1' bit is appended followed by '0'bits until the length of the message is congruent to 448 bits.

(2) This is followed by a 64-bit representation of the original message length. This technique is known as SHA-2 strengthening and overcomes security problems arising from messages of different lengths hashing to the same output. Padding is also done because 512 bits of data are manipulated at a time during the one-way hash process.

(3) The hashing process proceeds by taking the 512 bits of data at a time, combine the input with the 256 bit output of the previous round, and obtain the 256 bit representation. A known initialization vector (IV) is used at the start of the process.

(4) The total number of input bits to the feed forward network is 512 bit data +256 bit IV +1 bit bias = 769bits Hidden layer consisting 64 nodes, Output layer consisting of 256 nodes.

(5) The feed forward network can be represented in matrix form as

$$\begin{bmatrix} W_{1,b} & W_{1,0} & \cdots & W_{1,639} \\ W_{2,b} & W_{2,1} & \cdots & W_{2,639} \\ \vdots & & \ddots & \\ W_{64,b} & W_{64,1} & \cdots & W_{64,639} \end{bmatrix} \times \begin{bmatrix} 1 \\ i_0 \\ \vdots \\ i_{768} \end{bmatrix} = \begin{bmatrix} o_0 \\ o_1 \\ \vdots \\ o_{63} \end{bmatrix} \qquad (3)$$

$$\begin{bmatrix} W_{1,b} & W_{1,0} & \cdots & W_{1,63} \\ W_{2,b} & W_{2,1} & \cdots & W_{2,63} \\ \vdots & & \ddots & \\ W_{256,b} & W_{256,1} & \cdots & W_{256,63} \end{bmatrix} \times \begin{bmatrix} 1 \\ o_0 \\ \vdots \\ o_{639} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{255} \end{bmatrix} \qquad (4)$$

Where Y_0 to Y_{255} is the result of the output layer, O_0 to O_{63} is the result from the hidden layer and $W_{x,y}$ is the weight corresponding to neuron x in the layer and input y.

Take i, o, and y to be the array of input bits, hidden layer output bits and output layer output bits respectively.

$$\mathbf{O\ (i) = sigmoid\ (o\ (i),\ \alpha)}$$
$$\mathbf{Y\ (i) = sigmoid\ (y\ (i),\ \alpha)}$$

(6) Apply the piecewise linear chaotic map (pwl) for T times at each layer of each neuron of neural network. The chaotic map hash some properties suitable for constructing a cipher, such as initial-value sensitivity or parameter sensitivity. If the chaotic map iterated for T (T is big >50) times, slight difference in the initial value X(k) or the parameter Q causes large differences in the iterated value X(K+T). Generally, the chaotic function is iterated for T(T>50) times to keep the output randomness.

$$X(k+1)=f(X(k),Q)=\begin{cases} X(k)/Q, & 0\le X(k)<Q \\ (X(k)-Q)/(0.5-Q), & Q\le X(k)<0.5 \\ (1-Q-X(k))/(0.5-Q), & 0.5\le X(k)<1-Q \\ (1-X(k))/Q, & 1-Q\le X(k)\le 1 \end{cases} \quad (5)$$

Where Q is the control parameter and satisfies 0<Q<0.5

(7) In order to keep low cost, the map f() is iterated for only once for hidden layer and T times for output layer .

(8) Additially, the parameter W_i, B_i and Q_i change with the plain-block, and they are under the control of the previous Hash block H_{i-1}.

$$(W_i,B_i,q_i)=Pg(C_{i-1})=\begin{cases} q_i = Pwl(\sum_{j=0}^{3} c_{i-1,j}, q_{i-1}, 16) \\ b_{i,j} = Pwl(c_{i-1,j}, q_i, 16)\oplus b_{i-1,j} \\ w_{i,j,k} = Pwl(c_{i-1,j}, q_i, k)\oplus w_{i-1,j,k} \end{cases}$$

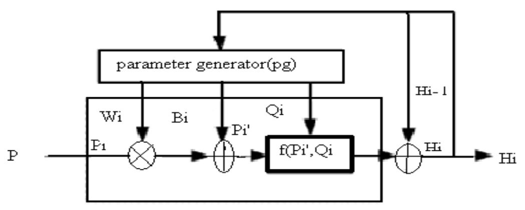


Figure 2: Hash function based on feed forward network with chaotic map

4. PERFORMANCE ANALYSIS

4.1 One-way property

In this Hash function, given a string of plaintext, it is easy to compute a hash value. But it is difficult to compute the plaintext if only the Hash value is known, also known as Pre-image resistance. This is important as one-way hash functions are used in digital signatures. Instead of signing a long message, which can take a long time, the hash of the long message is signed instead.

From equation 3 and 4, it seems possible to find o from y simply by performing Gauss-Jordan reduction. This can be followed by performing Gauss-Jordan reduction again to find the input i. luckily, the sigmoid function makes finding the unknown o and I extremely difficult. Using the knowledge that the output element of o lie between the range of 0 and 1 since it is the output of a previous sigmoid function, and by increasing the value of α in equation2, it is possible to modify the sigmoid function such that the probability of finding the inverse is extremely small. As α increase, the sigmoid function approaches that of a step function shown figure 6

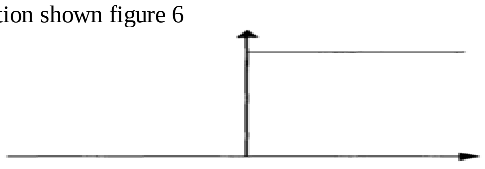


Figure 3: Step function

4.2 Hash results of message

Plaintext chosen is: " Passage 1: Earthquake and Tsunami hits Japan on March 12, 2011. Earthquakes are frequently occurring in Japan, approximately 195 events have been recorded. Tsunami destroyed the coastal area of Japan. Monday March 14, 2011, Three days after a powerful earthquake-triggered, Tsunami hits northeast Japan. Natural disasters like a severe Earthquake and Tsunami damaged Japan Fukushima daiichi nuclear plant. Tsunami destroyed the town in Sumatra, caused by the December 24, 2004 Indian Ocean Earthquake”.

C1:Change the number “1”in the original message into”3”.

C2:Change first letter “P” in the original message into “p”.

C3: Change the word “Earthquake” in the original message into “Tsunami”.

C4: Change the full stop at the end of the original message into comma.

The corresponding Hash values in Hexadecimal format are:

Original:
CABA647F16F185DF39A24E5AF75D38EA310D54BDB
8F1EB8C2BB9EB5368F8A4F1

C1:
3FE4490581D8E48526B4B618B9539F5254031500CD06
706230B5EA53F236EE0B

C2:
42855967DC13A359F083E52902A558599EB2502BE562
AF8652BAEC54FA13C3B3

C3:
DFA3E24927F285BBAFB564C528D929F9E1919BC2CE
4D034638BAEA5235F280A6

C4:
5F29242FE756C6A62457EF753B975869E1919BC2CE4
D034634BAE95D35F280A6

The simulation result indicates that the one-way property is so perfect that small change in the message will cause huge changes in the final hash value. The corresponding graphical display of binary sequences is shown in Fig.7

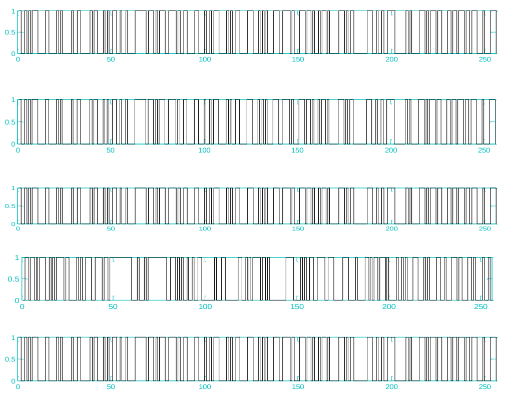


Figure 4: Hash value under different conditions

4.3. Plaintext Sensitivity:

For a hash function, it is required that different plaintext produce different hash values. This property depends on the hash function’s plaintext sensitivity. Slight difference in the plaintext will cause great changes in the Hash value, which makes the Hash function of high plaintext –sensitivity. This property is important to keep it secure against statistical attacks.

Experiments are done to test the hash function’s plaintext sensitivity. As an example, plaintext chosen is: " Passage

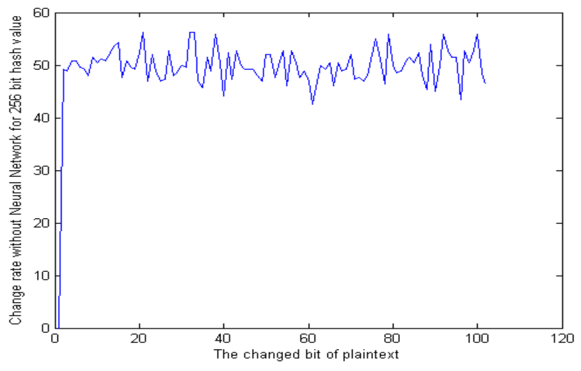
1: Earthquake and Tsunami hits Japan on March 12,2011.Earthquakes are frequently occurring in Japan, approximately 195 events have been recorded. Tsunami destroyed the coastal area of Japan. Monday March 14, 2011, Three days after a powerful earthquake-triggered, Tsunami hits northeast Japan. Natural disasters like a severe Earthquake and Tsunami damaged Japan Fukushima daiichi nuclear plant. Tsunami destroyed the town in Sumatra, caused by the December 24, 2004 Indian Ocean Earthquake".

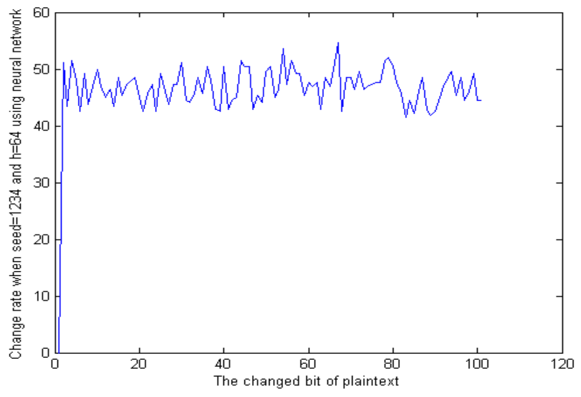
The according Hash value is H_o = A6CD7FF949F5B0B1CFF77DD2CA6F141D.Then only first bit of the plaintext is changed, the according hash value is H_1= D788A5A165919C6D5E9CC7396CE956EF.If the second one is changed, the according Hash value is, H_2=E5772AAB911F6EEAB39ED64E9369094B .And if the i-th one is changed ,then the according Hash value is H_i, Then the Hash value Change rate is computed by

$$(i)= \frac{\text{Dif } (H0,Hi) \ X\ 100}{256}$$

Where Dif (H_o, H_i) means the number of the different bits between H_o and H_i. For the proposed plaintext, the change rate is shown in Fig. 8. As can be seen, the change rate is about 50% (64 bits), which shows that the Hash function is of high plaintext sensitivity.

Plain text sensitivity graphs are as shown below in Figure 5.





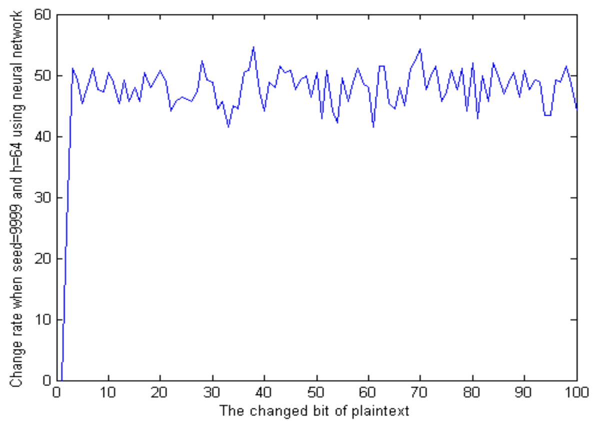


Figure 5: Plaintext sensitivity of SHA-2 Hash function

4.4 Collision Resistance

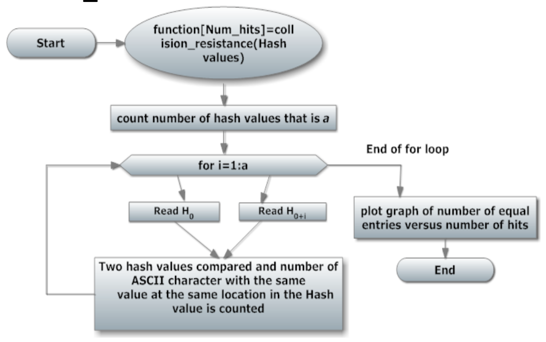

Start
function[Num_hits]=collision_resistance(Hash values)
count number of hash values that is a
End of for loop
for i=1:a
Read H_0
Read H_{0+i}
plot graph of number of equal entries versus number of hits
Two hash values compared and number of ASCII character with the same value at the same location in the Hash value is counted
End

Figure 6: flowchart for testing collision resistance of Hash values

Collision resistance is the difficulty of finding a different input that will map to the same output after hashing, for example H(M')=H(M).As the hash is supposed to provide a "unique" representation of the input data, allowing another disparate input data to hash to the same output defeats the purpose. Birthday attacks are similar in idea, expect that instead of finding another input that will hash to a required output, two random input data are found such that they hash to the same output. Birthday attack resistance is also known as 2^{nd} Pre-image resistance. Thus it should require a search though 2^{64} before two such inputs that hash to the same output are found. Flowchart for testing collision resistance of Hash values is shown in figure 6.

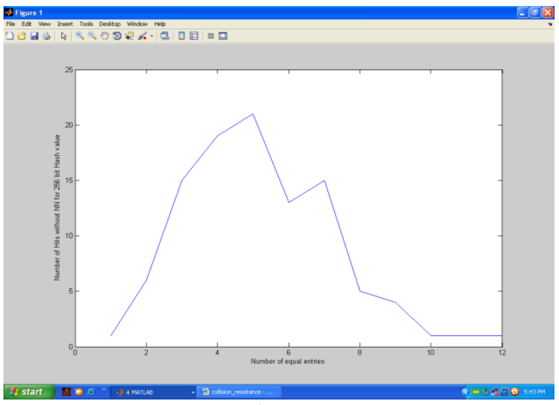


Figure 7: Screen shot of Collision Resistance graph without using neural network

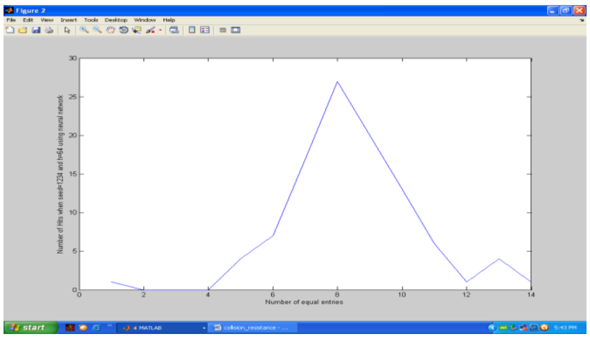


Figure 2
File Edit View Insert Tools Desktop Window Help
Number of Hits when seed=1234 and h=64 using neural network
Number of equal entries
start
4 MATLAB
collision_resistance - ...
5:43 PM

Figure 8: Screen shot of Collision Resistance graph for seed value= 9999 and number of hidden neurons 'h'=64

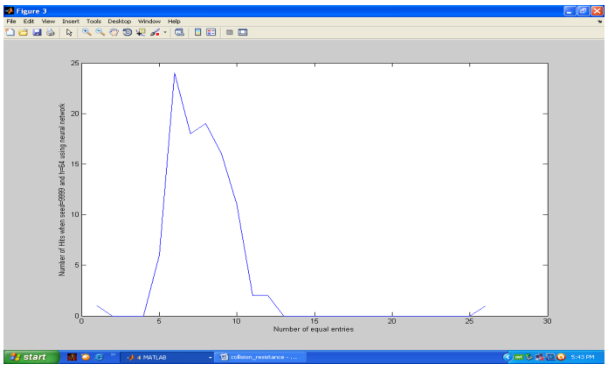
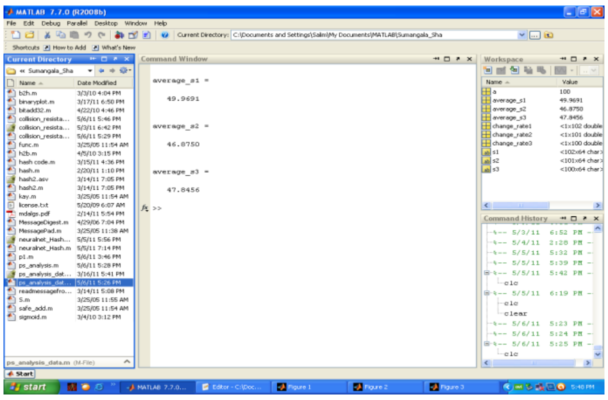


Figure 3
File Edit View Insert Tools Desktop Window Help
Number of Hits when seed=9999 and h=64 using neural network
Number of equal entries
start
4 MATLAB
collision_resistance - ...
5:43 PM

Figure 9. Screen shot of collision Resistance graph for seed value=1234 and number of hidden neuron 'h'=64


MATLAB 7.7.0 (R2008b)
File Edit Debug Parallel Desktop Window Help
Current Directory: C:\Documents and Settings\Salim\My Documents\MATLAB\Sumangala_Sha
Shortcuts How to Add What's New
Current Directory
Command Window
Workspace
Command History
start
MATLAB 7.7.0...
Editor - C:\Doc...
Figure 1
Figure 2
Figure 3
5:40 PM

Name	Date Modified
b2h.m	3/3/10 4:04 PM
binaryplot.m	3/17/11 6:50 PM
bitadd32.m	4/22/10 4:46 PM
collision_resista...	5/6/11 5:46 PM
collision_resista...	5/3/11 6:42 PM
collision_resista...	5/6/11 5:29 PM
func.m	3/25/05 11:54 AM
h2b.m	4/5/10 3:15 PM
hash code.m	3/15/11 4:36 PM
hash.m	2/20/11 1:10 PM
hash2.asv	3/14/11 7:05 PM
hash2.m	3/14/11 7:05 PM
kay.m	3/25/05 11:54 AM
license.txt	5/20/09 6:07 AM
mdalgs.pdf	2/14/11 5:54 PM
MessageDigest.m	4/29/06 7:04 PM
MessagePad.m	3/25/05 11:38 AM
neuralnet_Hash...	5/5/11 5:56 PM
neuralnet_Hash.m	5/5/11 7:14 PM
p1.m	5/6/11 3:46 PM
ps_analysis.m	5/6/11 5:28 PM
ps_analysis_dat...	3/16/11 5:41 PM
ps_analysis_dat...	5/6/11 5:26 PM
readmessagefro...	3/14/11 5:08 PM
S.m	3/25/05 11:55 AM
safe_add.m	3/25/05 11:54 AM
sigmoid.m	3/4/10 3:12 PM

ps_analysis_data.m (M-File)

```
average_s1 =

   49.9691


average_s2 =

   46.8750


average_s3 =

   47.8456

>>
```

Name	Value
a	100
average_s1	49.9691
average_s2	46.8750
average_s3	47.8456
change_rate1	<1x102 double
change_rate2	<1x101 double
change_rate3	<1x100 double
s1	<102x64 char>
s2	<101x64 char>
s3	<100x64 char>

```
-%-- 5/3/11   6:52 PM --
-%-- 5/4/11   2:28 PM --
-%-- 5/5/11   5:32 PM --
-%-- 5/5/11   5:39 PM --
-%-- 5/5/11   5:42 PM --
  clc
-%-- 5/5/11   6:19 PM --
  clc
  clear
-%-- 5/6/11   5:23 PM --
-%-- 5/6/11   5:24 PM --
-%-- 5/6/11   5:25 PM --
  clc
```

Figure 10. Average values of collision Resistance graphs screen shot

5. COMPARISON SHA-2 AND NEURAL NETWORK HASH

One of the properties such as plaintext sensitivity of neural network implemented hash function is compared against traditional hash function SHA-2 for the above plaintext. Consider above mentioned plaintext. The plaintext sensitivity graph for both SHA-2 and Neural Network Hash is plotted as shown in fig 9

From above simulation results, both SHA-2 and Feed forward neural network with chaotic map implemented Hash function gives on average same plaintext sensitivity.

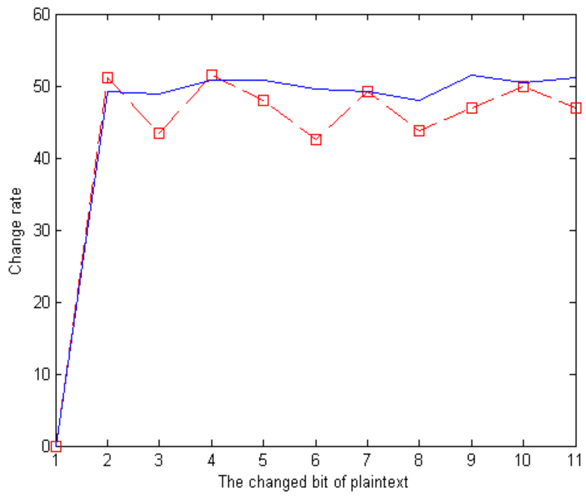


Figure 11. Screen shot comparison of performance graph

6. CONCLUSION

In this paper, an algorithm for one-way Hash function construction based on two layer feed forward neural network along with piece-wise linear chaotic map is presented. This hash function adopts the neural network's one-way property, diffusion property, and chaotic map confusion property. Applying a feed forward network as a hash algorithm presents several useful attributes. Firstly, simply changing the initial weight values of the feed forward network employed results in a totally different hash output. Many hash algorithms can be implemented for different purposes by setting unique initial weight values, for each purpose. Secondly, the feed forward network structure can be modified, allowing for a hashed key of more than 256 bits, simply by adding more neurons into each layer. The analysis and experiments shows that the change rate of plaintext sensitivity remains constant on an average 50% for different values of number of neurons in hidden layer and seed value which is used for generation of weights and biases at each layer of neural network. The collision resistance of SHA-2 is close to traditional algorithm.

7. REFERENCES

[1] L.P. Yee, D. L.C. Silva, "Application of Multilayer Perceptron Network as a One-way Hash Function" International Joint Conference on Neural Networks, Vol. 2,2002.

[2] Li, C., S. Li, D. Zhang and G. Chen,"Cryptanalysis of a chaotic neural network based multimedia encryption scheme". Advances in Multimedia Information Processing PCM ,2004 .

[3] Khalil Shihab, "A Backpropagation Neural Network for Computer Network Security" Journal of Computer Science 2 (9): 710-715, 2006.

[4] Shiguo Lian, Zhongxuan Liu, Zhen Ren, Haila Wang, "Hash Function Based on Chaotic Neural Networks" IEEE, 2006.

[5] Shiguo Lian,Jinsheng Sun,Zhiquan Wang, “ One-way Hash Function Based on Neural Network” Journal of Information Assurance and Security,2006

[6] Yi Du, Detang Lu, Daolun Li,“An Effective Hash-based Method for Generating Synthetic Well Log” 2008 IEEE.

[7] Qun-ting Yang,Tie-gang Gao,Li Fan,Qiao-lun Gu, “ Analysis of One-way Alterable Length Hash Function Based on Cell Neural Network” Fifth Intenational Conference on Information Assurance and Security,2009

[8] Qinghua Zhang,Han Zhang and Zhaohui Li,”One-way Hash Function Construction Based on Conservative Chaotic Systems” Journal of Information Assurance and Security, 5, pp.171-178, 2010

[9] Network Security Essential applications and Standards-William Stallings,Person Education,2000.

[10] R. Rivest: The MD5 Message Digest Algorithm. RFC 1321, 1992.

[11] Neural Network by Christos Stergiou and Dimitrios Siganos.