Hash Function Design Using Matrix Multiplication, Ex-or, Checksum Generation and Compression technique Approach

S.G.Srikantaswamy
Research Scholar, National Institute of Engineering, Mysore, Karnataka, India
Email: sg_srikantaswamy@yahoo.com
Dr.H.D.Phaneendra
Professor & Research Guide, National Institute of Engineering, Mysore, Karnataka, India
Email: hdphanee@yahoo.com

***Abstract*—Confidentiality and Authentication are the two important Services of Secured Communication. Authentication deals with the identification of proper source of information and data integrity. Hash function plays important role in providing data authentication. In our proposed scheme, we have suggested a method for developing hash function with traditional EX-OR operation and matrix multiplication, which makes the process irreversible. The Proposed scheme also involves the generation of checksum and padding it with the plaintext data. The hash value generated is subjected to run length coding compression technique to make the algorithm more robust. Thus the proposed scheme presents a fair technique for generation of hash function. It involves less coding, doesn't involve complex mathematical operations and hence works very fast and occupies less amount of memory.**

Keywords—Confidentiality; data integrity; Checksum; Hash function; Compression

I. INTRODUCTION

A hash function is any algorithm or subroutine that maps large data sets of variable length, called keys to smaller data sets of a fixed length[1].Hash functions resistant to collision attacks can be developed by combining the MD-5 and SHA-1 algorithms[2]. Generation of Non-Invertible matrices in GF(2) for practical one way hash algorithm has been suggested. The present method describe non-invertible matrix which can be used as multiplication matrix in Hill cipher technique for one way hash algorithm[3]. Hypothesis testing is the task of deciding which of two hypotheses $H_{o\ or}\ H_{1,}$ is true , when one is given the value of a random variable U[4]. Hill cipher requires the inverse of the matrix for decryption. The inverse of a matrix does not exists always and non-invertible matrices can be used to build hash functions [5]. Cryptographic hash functions can be built with random Latin squares and non-linear transformations. This scheme satisfies the properties of hash functions [6]. PUF can be a viable alternative to a digital key stored in on-chip non-volatile memory [7]. The four secure algorithms such as SHA-1, SHA-256, SHA-384, and SHA-512 are iterative, one way hash functions that can process a message to produce a condensed representations called hash or message digest [8]. MD5,SHA1and RIPEMD are the most commonly used message digest algorithms. A secure hash function MD-192 produces 192bit message digest and uses a modified message expansion mechanism [9]. Several ongoing projects are evaluating the efficiency of the SHA-3 candidate. Modified grostl-256 can take arbitrary length of input and gives 256 bits output [10]. A detailed view over the hash functions MD5,SHA-1 and RIPEMD-160 has been carried out[11]. A detailed review over the cryptographic hash functions has been carried out in[12]. Hash functions also called message digests and one-way encryption, are algorithms that use no key[13]. The MD6 Message-Digest algorithm is a cryptographic hash function. It uses a Merkle tree –like structure to allow for immense parallel computation of hashes for very long inputs[14]. SHA-3 originally known as Keccak is a cryptographic hash function. SHA-3 uses the sponge construction in which message blocks are XOred into the initial bits of the state, which is then invertibly permuted[15]. RIPEMD-160 is a 160-bit message digest algorithm. There exist 128,256,and 320-bit versions of this algorithm, called RIPEMD-128,RIPEMD-256,RIPEMD-320, respectively[16]. The hash function H can be applied to a block of data of any size. H produces a fixed-length output[17]. Both message authentication codes and digital signature schemes are used to ensure the integrity or authenticity of transmitted messages[18]. For each performance study, a set of performance criteria or matrix must be chosen[19]. Throughput and memory are the two important performance metrics. The proposed algorithm has been designed to suit the requirements of a good hash function algorithm. In addition to matrix multiplication operation, logical operations have been included to perform bit-wise manipulations . The compression operation namely run-length coding which is being used in the present method makes the algorithm non-linear. The proposed algorithm involves less complex operations and hence easily be implemented and the main aim of the proposed method development is the design of hash algorithm which consumes low memory requirement with appreciable throughput . The paper has been organized in a flexible manner. Section II describes the proposed algorithm with brief introduction about its working. Section III presents the complete description of

the algorithm and also describes its working for an arbitrary length of a message. Section IV explains the working of the algorithm with an example and discusses the results and presents an analysis about the output of the algorithm. Section V presents the strength and features of the proposed algorithm and also compares the main features of the algorithm comparing it with the existing hash algorithms. Section VI gives conclusion drawn from the above discussions and analysis. The last section presents the list of references.

II. PROPOSED HASH ALGORITHM

i) Read the Plaintext Block input
ii) Arrange the Plaintext Characters in the form of 8x7 matrix form and replace them by their ASCII-Decimal equivalent form. Fill-up the vacant Position with column-sum.
iii) Take the EX-OR of the Column-Elements
iv) The Resultant Value of Step-3 represents a single row of 8 elements
v) Determine the ASCII-Decimal Sum of the row elements and Substitute it as 9^{th} Element
vi) Arrange the 9-elements in the form of 3X3 Matrix form
vii) Consider a Non-Invertible Matrix of Order 3X3
viii) Multiply the Values of Step 6 and Step 7 to get the Final 3X3 hash-value
ix) Arrange the Matrix in the form of linear –array
x) Apply run-length coding compression technique on the value obtained by step 9 to get the final ASCII-Decimal value and concatenate the coded values
xi) Apply the above procedure for all blocks and concatenate the hash values obtained for each block
xii) Truncate the final hash value to 1024-bits to get the final hash value.
xiii) Append the Message with 1024-bit message digest value and transmit

III. DESCRIPTION OF THE ALGORITHM

Consider a Message for which hash code has to be generated. "This Message is very Secret, Store it in a separate file. Transfer the same amount of money to the account number 123. Send the order to the company A1B1C1D. This matter is Very Urgent."

The logic is to Consider a block of 45-characters at a time. Generate a message digest of 9 characters. Like this for an arbitrary length message generate the message digest and concatenate the message digests generated such that the final length of the message digest is equal to 1024-bits.

Now Consider the first 45 characters of the above message. " This Message is very Secret. Store it in a Separate file". Let the 45 characters be named as C1,C2,C3,C4…….C45.

Arrange the 45-characters in the Matrix form of the order 6 X 8.

C1	C2	C3	C4	C5	C6	C7	C8
C9	C10	C11	C12	C13	C14	C15	C16
C17	C18	C19	C20	C21	C22	C23	C24
C25	C26	C27	C28	C29	C30	C31	C32
C33	C34	C35	C36	C37	C38	C39	C40
C41	C42	C43	C44	C45	X1	X2	X3

Replace each character by their corresponding ASCII-Decimal equivalent values. The value of the element present at Sixth row and Sixth column(X1) is replaced by the sum of all elements above that number in that particular column.

X1=C6+C14+C22+C30=C38

The value of the character of sixth row and seventh column (X2) is replaced by the sum of all elements present above that number in that particular column.

X2=C7+C15+C23+C31+C39

The value of the character of sixth row and eighth column(X3) is replaced by the sum of all elements present above that number in that particular column.

X3=C8+C16+C24+C32+C40

Now let us calculate the following values P1,P2,P3,P4,P5,P6,P7,P8 and P9 as follows.

P1= C1⊕C9⊕C17⊕C25⊕C33⊕C41
P2= C2⊕C10⊕C18⊕C26⊕C34⊕C42
P3= C3⊕C11⊕C19⊕C27⊕C35⊕C43
P4= C4⊕C12⊕C20⊕C28⊕C36⊕C44
P5= C5⊕C13⊕C21⊕C29⊕C37⊕C45
P6= C6⊕C14⊕C22⊕C30⊕C38⊕X1
P7= C7⊕C15⊕C23⊕C31⊕C39⊕X2
P8= C8⊕C16⊕C24⊕C32⊕C40⊕X3
P9=P1+P2+P3+P4+P5+P6+P7+P8

Now Arrange the values of the plaintext values P1 to P9 in the form of 3 X 3 Matrix. Now consider a key matrix which is Singular and irreversible.

Let the key Matrix be, $K = \begin{matrix} K1 & K2 & K3 \\ K4 & K5 & K6 \\ K7 & K8 & K9 \end{matrix}$

Now Multiply the Processed Plaintext Matrix with the Key Matrix to get the Hash Values.

[K] X [P] = [H]

$$\begin{bmatrix} K1 & K2 & K3 \\ K4 & K5 & K6 \\ K7 & K8 & K9 \end{bmatrix} \mathbf{X} \begin{bmatrix} P1 & P2 & P3 \\ P4 & P5 & P6 \\ P7 & P8 & P9 \end{bmatrix} = \begin{bmatrix} H1 & H2 & H3 \\ H4 & H5 & H6 \\ H7 & H8 & H9 \end{bmatrix}$$

Concatenate the values of H1,H2,H3,H4,H5,H6,H7,H8 and H9 after applying run-length coding, we can get the Hash value for the given block of message. The same procedure is applied for the entire message , and hash value is obtained by concatenating the hash values obtained for each block. The hash value is truncated to 1024-bit to get final hash value for the given message. The above described method of generating 1024-bit message digest for a given message is efficient one since it includes multistage operations including bit-wise EX-OR, padding with check sum, and matrix Multiplication. The message can not be constructed

using the hash value since the key matrix used is a singular and irreversible matrix.

IV. RESULTS AND DISCUSSIONS

The logic here is to consider a Block of 45-characters at a time. Generate a message digest of 9 characters. Like this for an arbitrary length message, generate message digest and concatenate the message digests generated for all blocks at the end and truncate them to 1024-bit to get the final message digest value.

Consider the following Message for which hash code is to be generated.-" **This Message is Very secret. Store it in a separate file. Transfer the same amount of money to the account number 123. Send the order to the company A1B1C1D. This matter is Very urgent."**

Now consider the first 45-characters of the above message as a block. i.e" **This Message is very Secret. Store it in a Separate file.**" Let the characters be named asC1,C2,C3,C4,C5,C6,C7,C6,C9,C10,C11,C12,C13,C14, C15,C16,C17,C18,C19,C20…….C45.

Replace the above characters by their ASCII-Decimal equivalent and arranging them in the matrix form of 6 X 8, we get the following values.

84	104	108	115	77	101	115	115
97	103	101	105	115	118	101	114
121	115	101	99	114	101	116	115
116	111	114	101	105	116	105	110
97	115	101	12	97	114	97	116
101	102	105	108	101	550	534	570

Let us Calculate the values of P1,P2,P3,P4,P5,P6,P7,P8 and P9.

P1=84⊕97⊕121⊕116⊕97⊕101=60
P2=104⊕103⊕115⊕111⊕115⊕102=6
P3=108⊕101⊕101⊕114⊕101⊕105=18
P4=115⊕105⊕99⊕101⊕12⊕108=0
P5=77⊕115⊕114⊕105⊕97⊕101=33
P6=101⊕118⊕101⊕116⊕114⊕550=598
P7=115⊕101⊕116⊕105⊕97⊕534=636
P8=115⊕114⊕115⊕110⊕116⊕570=594
P9=60+6+18+0+33+598+636+594=1945

Arrange the above values in 3 X 3 matrix form.

60	6	18
0	33	598
636	594	1945

Consider a Key matrix of order 3 X 3, whose inverse does not exist. Let the singular Key matrix be K.

$$K = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}$$

Multiplying the altered plaintext matrix with the key matrix, we get

$$\begin{bmatrix} 60 & 6 & 18 \\ 0 & 33 & 598 \\ 636 & 594 & 1945 \end{bmatrix} \mathbf{X} \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} = \begin{bmatrix} 390 & 294 & 378 \\ 4318 & 4949 & 5580 \\ 16627 & 19802 & 22977 \end{bmatrix}$$

Thus the resultant hash Matrix H(M) is equal to:

$$H(M) = \begin{bmatrix} 00390 & 00294 & 00378 \\ 04318 & 04949 & 05580 \\ 16627 & 19802 & 22977 \end{bmatrix}$$

Applying the Mod-127 on the above Values, the resultant values are :

Resultant H(M)= H(M) Mod-127

$$\text{Resultant } H(M) = \begin{bmatrix} 9 & 40 & 124 \\ 0 & 123 & 119 \\ 117 & 117 & 117 \end{bmatrix}$$

Converting the above Matrix in one-dimensional array, we get

Resultant H(M)=9 40 124 0 123 119 117 117 117

Applying the Run-length Coding technique [20] on the above set of values, we get the following value. According to run length coding, 117 117 117 can be substituted by 117 ! 3 (i.e 117 has been repeated thrice, and the symbol ! indicates that 117 is a repeating character.

Thus the resultant Hash Value H (M)= 9 40 124 0 123 119 117 ! 3

Consider the next block of the message.i.e" **Transfer the same amount of money to the account numbe**. As discussed before , replace each character by their ASCII-Decimal equivalent and arrange them in the form of 6 X 8 matrix form, we get the following values.

84	114	97	110	115	102	101	114
116	104	101	115	97	109	101	97
109	111	117	110	116	111	102	109
111	110	101	121	116	111	116	104
101	97	99	99	111	117	110	116
110	117	109	98	101	X1	X2	X3

X1=C6+C14+C22+C30+C38
X2=C7+C15+C23+C31+C39
X3=C8+C16+C24+C32+C40

Therefore X1=550, X2=530 and X3=450.

Thus the above matrix can be represented as :

84	114	97	110	115	102	101	114
116	104	101	115	97	109	101	97
109	111	117	110	116	111	102	109
111	110	101	121	116	111	116	104
101	97	99	99	111	117	110	116
110	117	109	98	101	550	530	450

Now the values of P1, P2,P3,P4,P5,P6,P7,P8 and P9 are calculated as follows:

P1=84⊕116⊕109⊕111⊕101⊕110=41
P2=114⊕104⊕111⊕110⊕97⊕117=15
P3=97⊕101⊕117⊕101⊕99⊕109=26

P4=110⊕115⊕110⊕121⊕99⊕98=11
P5=115⊕97⊕116⊕116⊕111⊕101=24
P6=102⊕109⊕111⊕111⊕117⊕550=600
P7=101⊕101⊕102⊕116⊕110⊕530=622
P8=114⊕97⊕109⊕104⊕116⊕450=416
P9=41+15+26+11+24+600+622=416=1755
Arranging the above values in the form of 3 X 3 matrix form, we get:

41	15	26
11	24	600
622	416	1755

The above matrix represents the altered plaintext matrix. Multiplying the altered plaintext matrix with the key matrix yields the following values.

41	15	26		1	2	3
11	24	600	X	4	5	6
622	416	1755		7	8	9

The resultant hash value for the block under consideration is :

Resultant hash matrix =	283	365	447
	4307	4942	5577
	14571	17364	20157

Taking the Mod-127 of the above values, the resultant hash values are as follows:

Resultant hash matrix =	29	111	66
	116	116	116
	93	92	91

Converting the two dimensional matrix into linear array form, Then the resultant hash value becomes,
Resultant hash value=29 111 66 116 116 116 93 92 91
By applying the Run-length coding technique on the above set of values ,116 116 116 can be substituted by 116 ! 3 (i.e 116 has been repeated thrice, also ! indicates that 116 is a repeating character. Concatenating the above values , the resultant hash value for the current block=29 111 66 116 ! 3 93 92 91

The resultant hash value for the two consecutive blocks is :
H(M)= 9 40 124 0 123 119 117 ! 3 29 111 66 116 ! 3 93 92 91

The resultant hash values obtained for each block will be concatenated and finally truncated to 1024-bits to get the final hash value for the given message.

V. STRENGTH OF THE ALGORITHM

The proposed algorithm has been developed by keeping in view of providing efficient hash value with minimum memory requirement and high throughput. The proposed method occupies memory less than 5KB and works faster with execution time less than 10μS . The proposed method involves bit-wise exclusive-or operations which enhances the avalanche effect. The use of non-invertible matrix multiplication operation makes the algorithm one-way. The compression using run-length coding improved the collision resistant property. The padding of checksum helps to increase the confusion property of the algorithm. The truncation operation makes it infeasible to trace the hash value in reverse direction. Thus the algorithm has been designed to incorporate all resistive features to provide data security. The algorithm has been designed to produce 1024-bit hash value and hence the method is resistant against brute-force attack. The table 1 gives a comparison among various existing hash algorithms. The proposed algorithm has been named as SGSPP hash algorithm and here SGS stands for S G Srikantaswamy and PP stands for Professor Phaneendra.

TABLE 1: A COMPARISONOF SHA1, MD5, RIPEMD160 AND SGSPP

Algorithm	***Message Digest length***
SHA1	160
MD5	128
RIPEMD160	160
SGSPP	1024

VI. CONCLUSIONS

Hash algorithms are used to generate the hash or message digest of a message. The hash value of a message is used for the purpose of authentication and data integrity. Many hash algorithms MDx, SHA series, and RIPEMD and some more improved hash algorithms exist in literature. In our method, we have fused matrix multiplication, logical Ex-or , checksum padding operations to make the algorithm one way , and also to incorporate confusion property. The proposed algorithm can be applied to the message of any length and concatenated hash values of all blocks is truncated to get 1024-bit hash value. The use of compression algorithm like run-length coding improves the non-linear and collision resistant property. The proposed method can be further improved by using modular arithmetic, discrete logarithm operation and variable block length values.

REFERENCES

[1] http:en.wikipedia.org/Hash_function
[2] H.MirvaZiri, A new Hash Functions based on Combination of Existing digest algorithms, SCORed 2007, IEEE.org
[3] Artan Berisha,Behar baxhaku and Artan alidema, A class of Non Invertible Matrices in GF(2) for Practical One way Hash Algorithm, International Journal of Computer applications(0975-8887), Volume 54,No.18, September 2012
[4] Ueli M.Maurer,Authentication Theory and hypotheses Testing,IEEE Transactions on information
[5] Mohammed Abu Taha,Mousa Farajallah and Radwan Tahboub, A Practical One Way Hash Algorithm based on Matrix multiplication , International Journal of Computer Applications(0975-8887),Volume 23,No.2,June 2011
[6] Saibal K.Pal, Diwakar Bharadwaj, Rajat Kumar , Varun Bhatia, A new Cryptographic Hash Function based on Latin Squares and non-linear Transformations-2009,IEEE International Advance Computing Conference (IACC 2009),Patialia,India,6-7 March 2009
[7] Jae W.lee, Daihyun, Blaise Gassend, G.Edward suh, Marten van Dijk , Srinivas Devadas, A Technique to Build a Secret Key in Integrated Circuits for identification and authentication applications, 2004,Symposium on VLSI circuits digest of Technical papers, 0-7803-8287,2004 IEEE
[8] T.Lalitha, Dr.R.Umarani, Properties and approach of Cryptographic Hash algorithms , Indian Journal of Computer Science and engineering, Vo.1,No.1,58-65
[9] Harshvardhan Tiwari , Dr,Krishna Asawa, A Secure hash MD-192 with modified message Expansion, International Journal of Computer Science and information Security, Vol.VII,No.II, FEB2010
[10] Gurpreet Kaur,Vidyavati S nayak, Dhananjoy Dey, SK Pal, Analysis of Hash Function-Modified Grostl, International Journal of Computer applications(0975-8887), Volume 44, No.21, April 2012
[11] Harshvardhan Tiwari , Krishna Asawa, Cryptographic hash function:An Elevated View, European Journal of Scientific Research , vol.43, No.4(2010), pp 452-465
[12] Rajeev Sobti , G.Geetha, Cryptographic Hash Functions:A Review, International Journal of Computer Science issues (IJCSI), Vol.9, Issue 2, No.2, March 2012
[13] Nidhi Sharma , Alok Sharma and Monika Sharma, A More Secure Hashing Scheme for Information Authentication, National workshop-cum-Conference on recent trends in Mathematics and Computing(RTMC) 2011, Proceedings published in International Journal of Computer Applications (IJCA)
[14] http://en.wikipedia.org/wiki/MD6
[15] http://en.wikipedia.org/wiki/SHA3
[16] http://en.wikipedia.org/wiki/RIPEMD
[17] William Stallings, Cryptography and Network Security principles and practices, Third Edition, Peaeson Education
[18] Jonathan Katz , Yehuda Lindell, Introduction to Modern cryptography, Chapman & hall/CRC, taylor & Francis group, 2008
[19] Raj Jain, The art of Computer Systems Performance analysis, John Wiley & Sons, Inc.
[20] Ralf Steinmetz, Klara nahrstedt, Multimedia : Computing, communications & applications, Pearson Education, fifth Indian reprint 2004

AUTHORS PROFILE

Dr.H.D.Phaneendra has been working as a Professor at National Institute of Engineering, Mysore, Karnataka, India. He has more than 20 years of Teaching experience. He has guided more than 20 M.Tech Projects. He has published numerous research papers in national and international journals.

S.G.Srikantaswamy has been working as a Selection grade lecturer (TC), Department of Computer Science at JSS Polytechnic for the Differently Abled, Mysore, Karnataka, India. He has more than 15 years of teaching experience. He has been pursuing Ph.D under the guidance of Prof. H.D.Phaneendra at National Institute of Engineering, Mysore.