Implementation of a new 32-bit Algorithm for Password Hashing

[1]Rohan Dham
Department of Computer Science
GB Pant Engineering College
New Delhi ,India

, [2]Sahhil Tanwer,
Department of Computer Science
GB Pant Engineering College
New Delhi,India

[3]Anjali Arora
Department of Computer Science
GB Pant Engineering College
New Delhi,India

[4]Aashita Chhabra
Department of Computer Science
[4] WCTM,Gurgaon
New Delhi,India

***Abstract* – The importance of hash algorithms for protecting the authenticity of information is exhibited. Application includes probity protection, traditional message authentication and digital signatures. As it is evident from recent history that none of the hash algorithms are completely secure and have been broken one way or the another. The new algorithm described in this paper take some of the good features from MD5 and SHA1. It takes an input of 32-bit and produces a 64-bit hash as a result it makes the system more secure. Various techniques have been employed to induce randomness in the hash so that it makes the process more rigorous and provide fast execution on 32 bit machine.**

***Keywords:* Hash, Complexity, MD5, Cryptography, Password, IDEA**

I. INTRODUCTION

Hashing is a technique to map variable-length digital data to fixed-length digital data. Password hashing is primarily based on cryptographic hashing and assures privacy and security of sensitive data (in this case, the passwords) just like cryptographic hash algorithms do. The algorithm defined in below is a one-way hash algorithm that is it produces a hash for a password without any trail tracking back to the password (message). The motivation of this algorithm comes from MD5 and SHA1[1,6,7,8] cryptographic algorithm which paved the way to a more secure data storage and transportation.

Network Security & Cryptography is a concept to protect network and data transmission over wireless network. Data well-being is the main aspect of secure transmission of data over unreliable network.[2] Data protection remains challenging issue of data communications today that touches many areas including secure communication medium , validated data encryption technique and trusted third party to maintain the database.[3,4] The rapid development in information technology, the secure transmission of confidential data herewith gets a great deal of attention. The conventional methods of encryption can only maintain the data security. The information could be accessed by the unauthorized user for malicious purpose. Therefore, it is necessary to apply effective encryption/decryption methods to enhance data security [9, 10].

We Should keep in mind that hash functions are used to authenticate password that are not exactly the hash functions which are being used in data structure course .The implementation of hash functions used in a form of

hash tables are extremely fast but not secure .It is said that only cryptographic hash functions are used to implement password hashing . We do have hash functions like SHA256 ,SHA512, RipeMD and WHIRLPOOL . It's been seen that we have to run the password through a cryptographic hash function and then it will accessed easily. [10]

II. DESIGN CRITERIA OF NEW ALGORITHM

The initial work starts by taking a 32-word long password and an email address as input and their binary form is stored in blocks of ten bits per block. A series of shifting, embedding, padding and rotation introduces branching in the code which poses doubt in the minds of the hackers so that even if the database is compromised, the hacker won't have any way to create a pre-image from the hash forms. The number of blocks can be calculated by the formula given below:

$$k=\lceil n/(t-1)\rceil \qquad (1)$$

Where, k= Total number of blocks.

t= 11

n= total number of message bits

THE NEW ALGORITHM

INPUT: Variable length password and email with the maximum accepted length being 32-bit.

OUTPUT: 64-bit hash

MATHEMATICAL FORMULAE USED:

$$k=\lceil n/(t-1)\rceil$$

$$d=(k*(t-1))-n \qquad (2)$$

Step 1:

- Accept a password message from user.
- Store its ASCII form in an array password_ascii array.

In another array, store the main_message_aux array and store the binary form of the message.

Step 2:

- Next, we segregate the bits from the main_message_aux[ ] into blocks of 10's. The number of blocks (k) can be found using the formula given above.
- Padding is done to the last block to make it exactly divisible by 10 and to make integer number of blocks. The number of extra bits (d) to be padded can be calculated using the formula given above.
- To introduce randomness in the hash, we add two more blocks of bits to the total.
- The $(k+1)^{th}$ block contains the binary equivalent of the number of bits padded to the k^{th} block.
- And, the $(k+2)^{th}$ block contains the binary equivalent of the total number of 1's in the first k blocks. All of these are stored in a 2-D array

Step 3: Next, convert the 10-bit blocks to 64-bit blocks by "embedding" the 10 bits at specific positions and padding other positions with either 0's or 1's.

Step 4: The IV can be either predefined in the program, or the binary equivalent of the email could form the Initialization Vector (IV).

Step 5: The result of steps 3 and 4 are sent as inputs to the compression function. The operation taking place is:

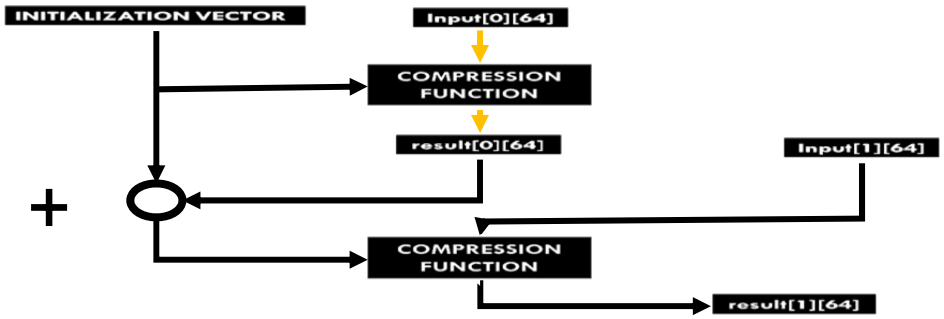

Fig 1 .It shows the compression function of an algorithm.

Step 6: The 64 bits are calculated as follows: 10 bits from step 2, 1 bit signifying whether the sum of ASCII bits of the input is even or odd and the rest of the bits are the latter 53 bits from the previous row of the 2-D array in which all this is stored.

Step 7: The compression function works as follows:

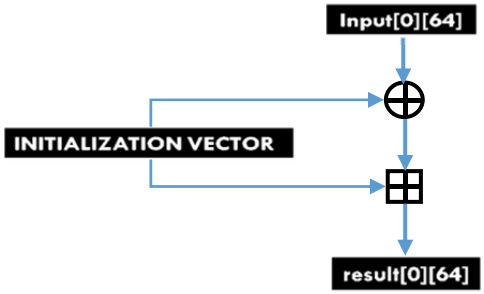

Fig2. It shows the working of compression function.

Step 8: The Prime Number Conundrum:

a. Calculate the first prime number greater than the ASCII sum of the input password and also a prime number just smaller than the ASCII sum.
b. Divide both these numbers with 32 and obtain the remainders
c. Right shift the first half of the resulting hash, obtained up till now, by the remainder of the smaller prime number obtained in the previous step. Similarly right shift the latter half of the hash by the second remainder.

III. IMPLEMENTATION

The algorithm takes as input a message of arbitrary length and produces as output a 64-bit "fingerprint" or "message digest" of that particular input. It is suspected that it is computationally infeasible to produce two messages having the same message digest, or to provide any message which already contains a given pre-specified target message digest. This algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key

cryptosystem such as RSA. The algorithm is designed to be quite fast on 32-bit machine. Further, it does not need any large substitution tables; the algorithm can be coded quite compactly.

IV. RESULTS

The first table shows the various inputs given to the algorithm and the outputs generated by the algorithm. As it is evident from the table that is if we provided various input to the algorithm it will generate different hash functions as well .It is also possible that without providing any input to the algorithm it still generate the hash function.

Test Cases	Hash Values
“”	**1112353365240578047**
1234567890	**1.46019313268E+19**
input data	**1.46019311268E+19**
abcdefghijklmnopqrstuvwxyz	**1957318365662685640**
ABCDEFGHIJKLMNOPQRSTUVWXYZ	**1.57203188086E+19**

Table 1: New Algorithm Execution

Total bit length graph shows the bit length of the final hash produced by us.
Our algorithm will be working over 32 characters input and will be producing 64 bit binary hash (in the form of 0's and 1's) which is further converted into 8 bit hash by taking the ascii values of 8 bits block at a time (for ex : 00100001 = 65 and ascii form of 65 is a).

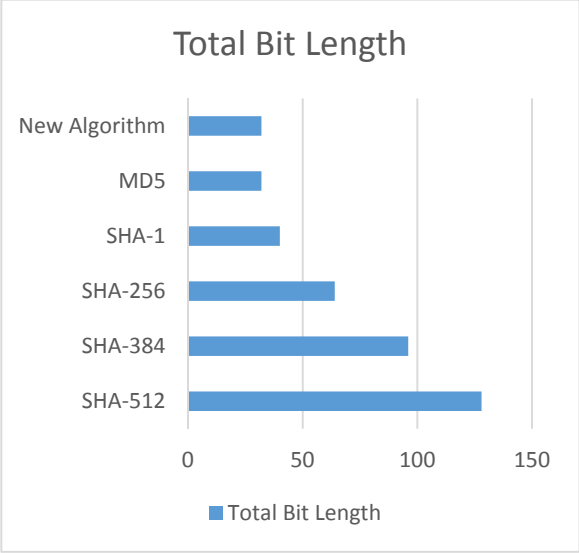

Table 2: Total Bit Length

The figure shows that our new algorithm is faster as compare to the conventional algorithms. The parsing time of the new algorithm is 0.031 seconds

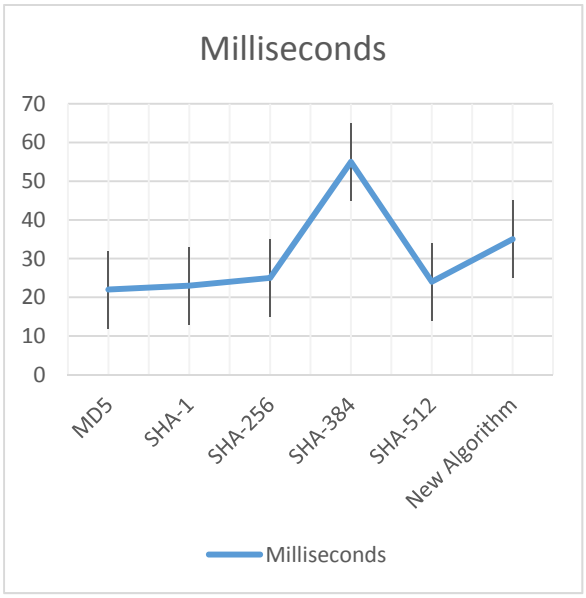

Table 3: Performance chart of hashing algorithms.

V. CONCLUSION

In this paper a new algorithm is described which may be light-weight, but produces quite startling results using small transformation methodology. A small executable test is being provided to display how the algorithm works, with the performance comparison with various hashing algorithm. This algorithm could be used together with an encryption algorithm to further reinforce it and send "compressed" data (hash) over a VPN for safer communication of sensitive data.

VI. REFERENCES

[1]. Rivest R., 1992, "The MD5 Message-Digest Algorithm,"RFC 1321, MIT LCS and RSA Data Security, and Inc.
[2]. Kahate, Atul, 2003, "Cryptography and Network Security", Tata McGraw-Hill, India.
[3]. Simmonds, A; Sandilands, P; van Ekert, L (2004) Ontology for Network Security Attacks". Lecture Notes in Computer Science. Lecture Notes in Computer Science 3285, pp.317–323
[4]. Kasgar A. K., Agrawal Jitendra, Sahu Santosh, 2012, "New Modified 256-bit MD5 Algorithm with SHA Compression Function", IJCA (0975–8887) Volume 42 (12) pp47-51.
[5]. William Stallings, Cryptography and Network Security: Principles and Practice, 5th Edition Prentice Hall; 5 Edition (January 24, 2010).
[6]. Vandana P., V.K Mishra, Architecture based on MD5 and MD5-512Bit Applications, IJCA (0975 – 8887) Vol. 74– No.9, July 2013.
[7]. Priyanka Walia, Vivek Thapar, 2014, Implementation of New Modified MD5-512 bit Algorithm for Cryptography
[8]. Piyush Gupta, Sandeep Kumar, 2014, A Comparative Analysis of SHA and MD5 Algorithm
[9]. Quynh Dang, 2009, Randomized Hashing for Digital Signatures
[10]. Elkamchouchi, H.M; Emarah, A.-A.M; Hagras, E.A.A, ―A New Secure Hash Dynamic Structure Algorithm (SHDSA) for Public Key Digital Signature Schemes‖, the 23rd National Radio Science Conference (NRSC 2006).

AUTHORS PROFILE

[1] Rohan Dham is a student in G B Pant Engineering College, Delhi and pursuing the last year of Bachelor of Technology degree in the Computer Science Engineering stream. He had worked on few projects till date which include a minor project on the Simulation of Networking **&** Routing protocols using OPNET and a seminar on VoIP and its Simulation

[2] Sahhil Tanwer is a student in G B Pant Engineering College, Delhi and am pursuing the last year of my Bachelor of Technology degree in the Computer Science Engineering stream.He had worked on a few projects

till date which include a minor project on the Simulation of Networking & Routing protocols using OPNET and a seminar on VoIP and its Simulation

[3]Ms. Anjali Arora is working as an Assistant Professor in G. B. Pant Engineering College, Delhi, India. She completed her M.Tech. from Banasthali University, Jaipur in 2012. Her Area of interest are Algorithms, Data warehouse & mining, DBMS, Computer Networks. She has published various papers in journals and conferences

[4] Aashita Chhabra is a student in WCTM ,Gurgaon and pursuing the last year of Masters Of Technology degree in Computer Science Engineering stream .She had worked on few projects till date which include a minor project on the Chat Server using Java and a major project on Steganography using .net C# . She had done internship program from a CMC Ltd Delhi .