A New Approach 160-bit Message Digest Algorithm

Priteshwar Nath Sallam
School of IT, Rajiv Gandhi Technical University Bhopal, (M.P) India

Jitendra Agrawal
School of IT, Rajiv Gandhi Technical University Bhopal, (M.P) India

Santosh Sahu
School of IT, Rajiv Gandhi Technical University Bhopal, (M.P) India

ABSTRACT
This paper introduces a new approach of MD Algorithm for security design. This approach comprises of the 160-bit hash algorithm for secure message digest. The results show that the 160-bit Message Digest Algorithm code is more secured than 128-bit Message Digest Algorithm code. This effort can alternate efficiently the accessible Message Digest Algorithm and hash function implementations for security design, so this approach proves a high security step towards design.

Keywords
MD Algorithm; Hash Function; Data Integrity;

1. INTRODUCTION
The software is designed for security purpose. A hash value computes a permanent length output called the message digest from an input message of different lengths. The MD5 message digest algorithm was developed by Ron Rivest at MIT. Until the last few years when both burst force and cryptanalytic concerns have arose , MD5 was most widely used secure hash algorithm. It is a widely-used 128-bit hash function, used in various applications Including SSL/TLS, IPSec, and many other cryptographic protocols. The MD5 algorithm breaks a sleeve into 512 bit input blocks. Every block is run from side to side a series of functions to produce a exceptional bit hash value for the sleeve[1].This paper explain how to design MD 160 algorithm and how MD 160 is more secure than past 128 bit hash algorithms.

2. HASH FUNCTION
A hash function H is a transformation that takes a variable-size input m and proceeds a fixed-size string, which is called the hash value h .Hash functions with just this property have a variety of general computational uses, but when working in cryptography the hash functions are regular chosen to have some supplementary properties. This is a contract in lots of programming languages that allocate the user to dominate equality and hash functions for an object, that if two objects are the same their hash codes must be the same. Hash functions compress a n (arbitrarily) large number of bits into a small number of bits.

The hash function properties are:-

- Output does not reveal information on input.
- Hard to find collisions (different messages with same hash).
- One way cannot be reversed.

3. STRENGTH OF MD 160
Every hash function with more inputs than outputs will essentially have collisions. This hash function MD-160 that generate 160 bits of output from an randomly large input. Since it must generate one of 2^{160} outputs for each member of a much larger set of inputs, the pigeon hole opinion guarantees that some inputs will hash to the same output. Collision conflict doesn't mean that no collisions exist; simply that they are hard to find. The birthday "paradox" spaces an upper bound on collision conflict: if a hash function generate N bits of output, an attacker who computes "only" 2N/2 hash operations on arbitrary input is likely to find two matching outputs. If there is an easier method than this brute force attack, it is typically measured a flaw in the hash function. Cryptographic hash functions are usually considered to be collision resistant. But many hash functions that were once thought to be collision resistant were later broken. MD5 and SHA-1 in exacting both have published approaches more efficient than brute force for pronouncement collisions. However, some compression functions have a proof that pronouncement collisions are at least as difficult as some hard mathematical problem (such as integer factorization or discrete logarithm). Those functions are called provably secure.

4. DATA INTEGRITY
Data integrity assertion and data basis authentication are important security services in an economic statement, electronic business, electronic mail, software distribution, data storeroom and so on[3]. The messages are broadest of verification within computing systems Encompasses uniqueness authentication, meaning source Authentication and message contented authentication. One that deals with individual message only without regard to large context generally provides protection against message modification only. Data integrity is compulsory within a record at its design steps. from beginning to the end use of standard rules and procedures, and is maintained through the use of error inspection and validation routines[3].

Data integrity divided are four categories:-

- Entity integrity
- Domain integrity
- Referential integrity
- User-defined integrity

4.1 Entity integrity
Entity Integrity ensures that there are no replica reports within the table and that the field that identifies each documents within the table is matchless and never null.

4.2 Domain integrity
Domain Integrity mostly used for business rules. A domain is the position of all probable values for a given element. A domain integrity check usually just called a domain constraint is a rule that defines these legal values.

4.3 Referential integrity
Referential integrity is a main property of data which are required for one attribute. Referential integrity is a record

thought that ensures that associations between tables remain consistent. Referential integrity works for RDBMS.

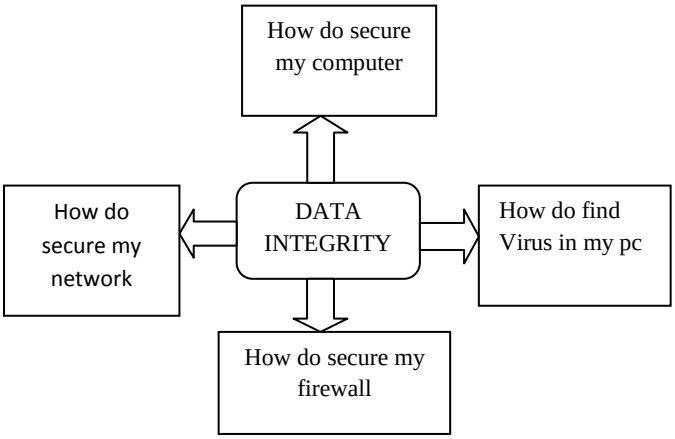

Fig4: Data Integrity System

4.4 User-defined integrity

User Define Integrity most commonly define by user that involve in the business. User-defined integrity allows you to define precise business rules that do not descend into one of the other integrity categories.

5. MD 160 BIT ALGORITHM

MD 160 is an algorithm that is used to authentic data integrity from side to side the creation of a 160-bit message digest from data input that is claimed to be as exceptional to that explicit data as a fingerprint is to the explicit person. This algorithm is designed on the base of MD5 that was designed by R. Rivest in 1991. In 1993 some weaknesses in it's propose be critical exposed. In 2004 meticulous Wang et al. generated a collision for MD5, accessible at Euro Crypt 2005 in [4]. Authors in [5] and [6] then extended this collision construction method, leading to the surprising result presented in [7] at the beginning of 2009. The authors devised a convenient attack situation, where they successfully created a scoundrel digital identity certificate issued by an unconscious real-world Certification Authority (CA) and trusted by all common web browsers. The MD5 hash function was developed in 1994 developed by Professor Ronald L. Rivest of MIT, is intended for use with digital signature .Changing for one digit in any of the input blocks should have a cascading produce that finally alters the hash results[2]. MD5 algorithm is to produce a message digest of information in organize to prevent tampering. We view the entire file as a large text message, and result in a unique MD5 message digest by the irreversible string alteration method. In the future, if the stuffing of file are changed, we only recalculate MD5 message digest of this categorizer and will find the difference from the original message digest. MD5 is one of the most admired hash functions for many applications such as IPsec [9].we have enthusiastic comparison with MD5, we arbitrarily change 1 to 1024 bits in plain text in that order, and for every adjust we repeat our algorithm and MD5 each for 20 time and record the normal rate of difference in message digests. The authentication does not need to order the original data but only need to have a valuable digest to authenticate the identity of client. This reduces the prospect that original data grasped by the intruder significantly and guarantees the information security. Let's See it design from 160 bit message digest algorithm:-

5.1 Padding Bits

The message is "padded" (extended) so that its length (in bits) is congruent to 448, modulo 512. That is, the message is extended so that it is just 64 bits shy of being a multiple of 512 bits long. Padding is always performed, even if the length of the message is already congruent to 448, modulo 512.Padding is performed as follows: a single "1" bit is appended to the message, and then "0" bits are appended so that the length in bits of the padded message becomes congruent to 448, modulo 512. In all, at least one bit and at most 512 bits are appended [11].

5.2 Append length

A 64-bit representation of b (the length of the message before the padding bits were added) is appended to the result of the previous step. In the unlikely event that b is greater than 2^64, then only the low-order 64 bits of b are used [11].

5.3 Initialize MD Buffer

A Five-word buffer (A, B, C, D, E) is used to compute the message digest. Here each of A, B, C, D, E is a 32-bit register. These registers are initialized to the following values in hexadecimal, low-order bytes first.

Word A: 01 23 45 67
Word B: 89 ab cd ef
Word C: fe dc ba 98
Word D: 76 54 32 10
Word E: c3 d2 d1 f0

This step uses a 80-element table T[1 ... 80] constructed from the sine function.
Do the following:

```
  /* Process each 16-word block. */
  For i = 0 to N/16-1 do
   /* Copy block i into X. */
   For j = 0 to 20 do
     Set X[j] to M[i*16+j].
    end /* of loop on j */
/* Save A as AA, B as BB, C as CC, and D as DD,E as EE. */
    AA = A
    BB = B
    CC = C
    DD = D
    EE = E
    /* Round 1. */
    /* Let [abcd k s i] denote the operation
         a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
    /* Do the following 20 operations. *
     [ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3]
[BCDA 3 22 4]
      [ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7
22 8]
      [ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11]
[BCDA 11 22 12]
      [ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15]
[BCDA 15 22 16]
      [ABCE 16 7 17] [EABC 17 12 18] [CEAB 18 17 19]
[BCEA 19 22 20]

  /* Round 2. */
   /* Let [abcd k s i] denote the operation
        a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
   /* Do the following 20 operations. */
```

[ABCD 1 5 21] [DABC 6 9 22] [CDAB 11 14 23] [BCDA 0 20 24]
[ABCD 5 5 25] [DABC 10 9 26] [CDAB 15 14 27] [BCDA 4 20 28]
[ABCD 9 5 29] [DABC 14 9 30] [CDAB 3 14 31] [BCDA 8 20 32]
[ABCD 13 5 33] [DABC 2 9 34] [CDAB 7 14 35] [BCDA 12 20 36]
[ABCE 1 5 37] [EABC 6 9 38] [CEAB 11 14 39] [BCEA 0 20 40]

/* Round 3. */
/* Let [abcd k s t] denote the operation
a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
/* Do the following 20 operations. */
[ABCD 5 4 41] [DABC 8 11 42] [CDAB 11 16 43] [BCDA 14 23 44]
[ABCD 1 4 45] [DABC 4 11 46] [CDAB 7 16 47] [BCDA 10 23 48]
[ABCD 13 4 49] [DABC 0 11 50] [CDAB 3 16 51] [BCDA 6 23 52]
[ABCD 9 4 53] [DABC 12 11 54] [CDAB 15 16 55] [BCDA 2 23 56]
[ABCE 5 4 57] [EABC 8 11 58] [CEAB 11 16 59] [BCEA 14 23 60]

/* Round 4. */
/* Let [abcd k s t] denote the operation
a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
/* Do the following 20 operations. */

[ABCD 0 6 61] [DABC 7 10 62] [CDAB 14 15 63] [BCDA 5 21 64]
[ABCD 12 6 65] [DABC 3 10 66] [CDAB 10 15 67] [BCDA 1 21 68]
[ABCD 8 6 69] [DABC 15 10 70] [CDAB 6 15 71] [BCDA 13 21 72]
[ABCD 4 6 73] [DABC 11 10 74] [CDAB 2 15 75] [BCDA 9 21 76]
[ABCE 0 6 77] [EABC 7 10 78] [CEAB 14 15 79] [BCEA 5 21 80]

/* Then perform the following additions. (That is increment each of the four registers by the value it had before this block was started.) */
A = A + AA
B = B + BB
C = C + CC
D = D + DD
E = E + EE
end /* of loop on i */

5.4 Process Message in 16-Word Blocks

This is the heart of the algorithm, which includes four "rounds" of processing. We take as input three 32-bit words and produce as output one 32-bit word [11]

F(X,Y,Z) = XY v not(X) Z
G(X,Y,Z) = XZ v Y not(Z)
H(X,Y,Z) = X xor Y xor Z
I(X,Y,Z) = Y xor (X v not(Z))

5.5 Outputs

The *L* 512-bit blocks have been processed, the output from L^th age is the 160-bit message digest.

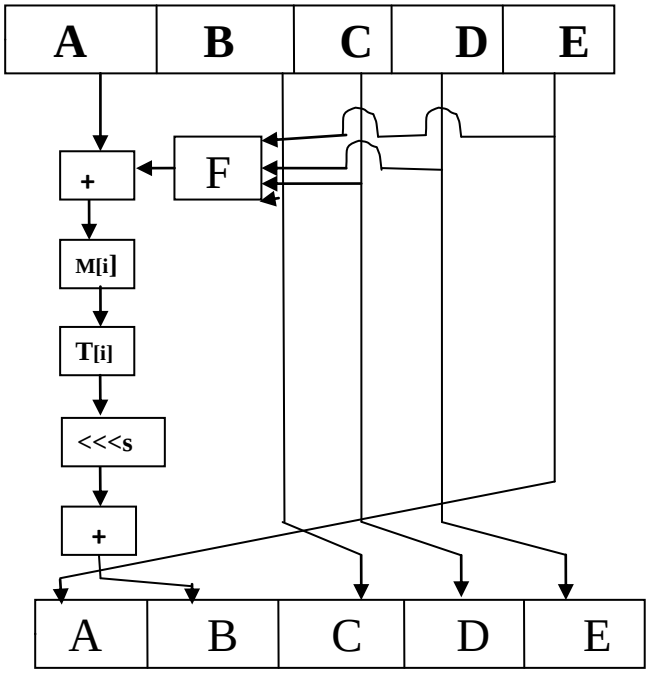

Fig 5.5: MD 160 output

6. RESULT

Hence when 160 bit is used in place of 128 bit Message Digest, collision is reduced and security is improved. Even this scheme is of 160 bits and need 280 bits for birthday paradox but it is strong enough for the first and second pre image attack. We can extend the length of hash to 256 or 512 bits to be more resistible against birthday attack which compared with its ancestors, is more powerful. In cryptography, a brute force attack or exhaustive key search is a strategy that can be used against in any encrypted data [15] theoretically by an attacker who is unable to take advantage of any flaws in an encryption system that would otherwise make his/her task easier.

The analysis of string by cryptool and comparison is done sequentially by the tools like entropy, floating frequency, histogram, n-gram, autocorrelation that are explained in detail in the following paragraphs.

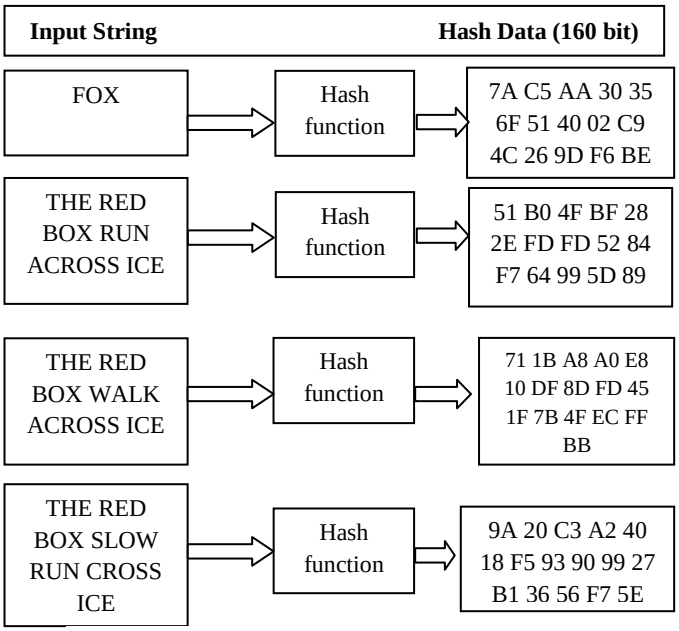

Fig 5.0: Conversion of hash value

6.1 Entropy

The entropy of a source thus indicates its characteristic distribution. It measures the average amount of information which one can obtain through observation of the source or, conversely, the indeterminacy which prevails over the generated messages when one cannot observe the source.

Table 1 Calculate the entropy

S.N	String	Entropy (maximum possible entropy 4.70)
1	FOX	1.58
2	THE RED BOX RUN ACROSS ICE	3.65
3	THE RED BOX WALK ACROSS ICE	3.78
4	THE RED BOX SLOW RUN ACROSS ICE	3.80

6.2 Floating Frequency

The floating frequency is the character which occurs most frequently. The input must contain at least 64 character(s) from the current alphabet. Calculate the all string Table 1.

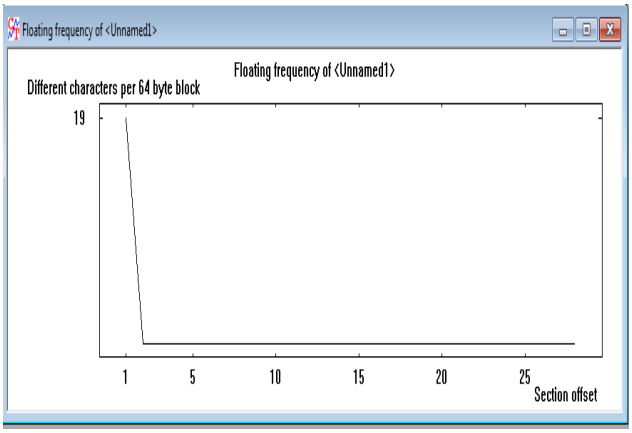

Fig 6.2: Floating frequency

6.3 Histogram

A histogram is one of the basic quality tools. It is used to graphically summarize and display the distribution and variation of a process data set. The main purpose of a histogram is to clarify the presentation of data.

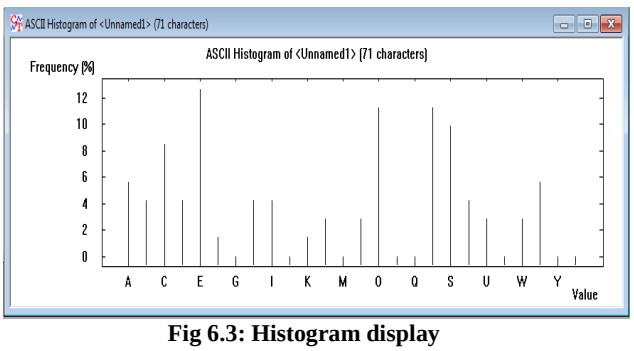

Fig 6.3: Histogram display

6.4 N-Gram

N-grams are essential in any task in which we have to identify words in noisy, ambiguous input. *N*-gram is a adjacent sequence of *n* items from a given sequence of text or speech.

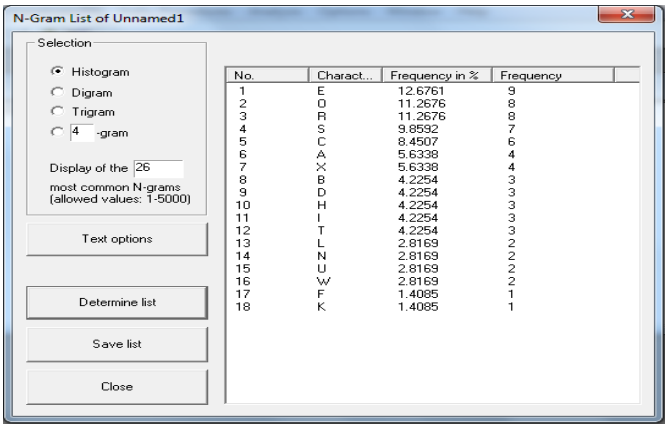

Fig 6.4: N-gram

6.5 Auto correction

This auto correction functionality is usually implemented by finding the string that is most similar to the given search string. The difference between two strings or minimum operation required to transform one string to another string is called edit distance between two strings. So edit distance is calculated between each string in database and given search string, the string having minimum edit distance cost is selected for the results.

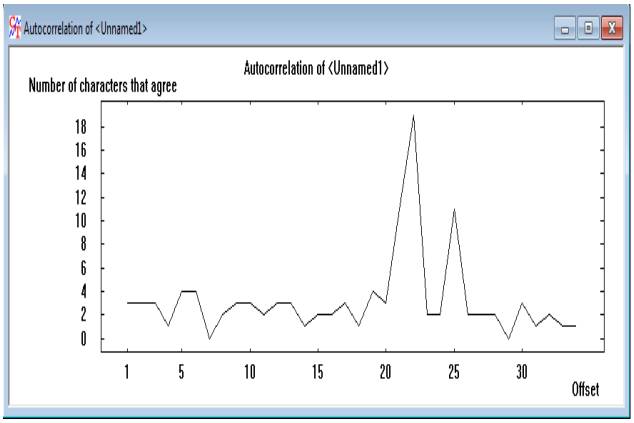

Fig 6.5 Auto correction

7. CONCLUSION

The simplified MD5 message-digest algorithm is simple to implement, and provides a ”fingerprint” or message digest of a message of random length. It is conjectured that the complexity of two messages having the same message digest is of the order of 2ˆ64 operations, and that the complexity of any message having a given message digest is of the order of 2ˆ160 operations. The Message Digest 128 bit algorithm is slightly cheaper to compute, however Message Digest Algorithm 128 is currently very vulnerable to collision attacks. Message digest compresses any stream of bytes into a 160 bit value. This compression goes only in one dimension. If you give the hash of a random stream of bytes to someone, there is no theoretical way for them to go back to unique stream of bytes.

7. REFERENCES

[1] R.Rivest. The MD5 Message-Digest Algorithm [rfc1321]

[2] Janaka Deepakumara, Howard M. Heys and R. Venkatesan, FPGA IMPLEMENTATION OF MD5 HASH ALGORITHM, FacuIq of Engineering and Applied Science Memorial University of Newfoundland St. John S, NF, Canada.

[3] Danyang Cao, Bingru Yang, Design and implementation for MD5-based data integrity checking system, College of Information Engineering, North China University of Technology , NCUT Beijing 100144, China

[4] X. Wang and H. Yu, "How to Break MD5 and Other Hash Functions", in: Ronald Cramer (ed.), *Advances in Cryptology - EUROCRYPT 2005*, vol. 3494 of Lecture Notes in Computer Science, pages 19-35, Springer,2005.

[5] M. Stevens, "On collisions for MD5", MSc Thesis,Eindhoven University of Technology, June 2007

[6] M. Stevens, A. Lenstra and B. de Weger "Chosen-prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities", in Moni Naor (eds), Science, pages 1-22, Springer Verlag, Berlin, 2007.

[7] A. Sotirov, M. Stevens, J. proceedings of *Advance in Cryptology - EUROCRYPT 2007*, vol. 4515 of Lecture Notes in Computer Appelbaum, A. Lenstra, D. Molnar, D.A. Osvik, B. de Weger, "MD5 considered harmful today: Creating a rogu CA certificate", 2009.

[8] Kimmo Järvinen, Matti Tommiska and Jorma Skyttä, Hardware Implementation Analysis of the MD5 Hash Algorithm, Helsinki University of Technology Signal Processing Laboratory Shenyang, China Otakaari 5 A, FIN-02150, Finland

[9] "IP Security Protocol (ipsec)," http://www.ietf.org/html.charters/ipsec-charter.html

[10] Janaka Deepakumara, Howard M. Heys and R. Venkatesan, FPGA IMPLEMENTATION OF MD5 HASH ALGORITHM, FacuIq of Engineering and Applied Science Memorial University of Newfoundland St. John S, NF, CanadaA I B 3x.

[11] MD5 Algorithm Description, http://www.cotse.com/CIE/RFC/1321/7.htm.

[12] Goldwasser, S. and Bellare, M. "Lecture Notes on Cryptography". Summer course on cryptography, MIT, 1996-2001

[13] M.M.J. Stevens (June 2007). On Collisions forMD5. http://www.win.tue.nl/hashclash/On%20Collisions%20for%20MD5%20-%20M.M.J.%20

[14] Xiaoyun Wang, Yiquin Lisa Yin, Hongobo Yu. Finding Collisions in the Full SHA-1. http://people.csail.mit.edu/yiqun/SHA1AttackProceedingVersion

[15] Christof Paar, Jan Pelzl, Bart Preneel (2010). Understanding Cryptography: A Textbook for Students and Practitioners. Springer. p. 7. ISBN 3642041000