Design of New Hash Algorithm with Integration of Key Based on the Review of Standard Hash Algorithms

Nidhi Garg
M.Tech Scholar
Echelon Institute of Technology, Faridabad, Haryana, India

Neeta Wadhwa
Asst. Professor
Echelon Institute of Technology, Faridabad, Haryana, India

ABSTRACT
Cryptographic Hash Functions are main building block of message integrity. These functions have many information security applications such as Digital Signatures, Message Authentication, Data Integrity and Key derivation. This paper presents comparative study of standard hash algorithms (MD5, SHA-1 and SHA-2), their security aspects and recent attacks. In this paper, a new hash algorithm with integration of key is proposed. This key is a shared secret key 'KEY' of 192-bit. The proposed algorithm produces a hash code of 192 bits from an arbitrary length input and serves the requirement of both the message integrity as well as source authentication. This algorithm consists of very simple steps, therefore would have lesser overhead and complexity as compared to the standard hash algorithms.

Keywords
Data Integrity, Hash Algorithms, MD5, SHA-1, SHA-2, pre-image, collision, Security

1. INTRODUCTION
Cryptographic Hash Functions play an important role in the world of Cryptography. Hash function takes a message of arbitrary length input and produces small but of fixed length output. Generally Hash functions are classified as Keyless or Keyed [1]. Keyless hash functions accept a variable length message M and produce a fixed length hash value H. On the other hand Keyed hash functions accept both a variable length message M and a fixed length key K to produce a fixed length hash value H_k. Keyless hash functions provide data integrity and Keyed hash functions provide both the data integrity as well as source authentication. Use of key for hash generation is known as Message Authentication Code (MAC).

1.1 Security Properties of Hash Function
A Cryptographic hash function should possess following three properties: [1]

a) *Pre-image Resistance:* A Hash Function H is pre-image resistant ,if it should be computationally infeasible to retrieve the original message from which the hash value was obtained i.e. from given hash value H(M), it is impossible to find original message M.

b) *Second Pre-image Resistance:* A Hash Function H is called Second Pre-image resistant if given a message M, it is impossible to find another message M' such that H(M)=H(M').

c) *Collision Resistance:* A Hash Function H is Collision Resistant, if it should be computationally infeasible to find any two messages M and M' such that H(M)=H(M') while M≠M'. For a secure hash function, the best attack to find a collision should not be better than the *birthday attack* [9] (i.e. not better than work complexity of $2^{n/2}$ for a hash function outputting n-bit hash values)

2. RELATED STUDY
2.1. Comparative Study of Existing Hash Functions (MD5, SHA-1 and SHA-2)
MD5 Hash Function: MD5[13] is a hash function designed by Ronald L. Rivest in 1992 as a strengthened version of MD4[14]. From an arbitrary length input message, the MD5 produces a single output of 128-bit message digest. The input message is composed of multiple blocks each of 512 bits. Rivest made following changes to MD4 to obtain MD5 were as follows:

i) Addition of fourth round of 16 steps and a Round 4 function.

ii) Replacement of Round 2 function by a new function.

iii) Modification of the access order for message words in Round 2 and 3.

iv)Addition of output from the previous step into each of 64 steps.

v) Modification of shift amounts (such that shifts differ in distinct rounds)

Attacks on MD5

In 1993, B. Den Boer and A. Bosselaers[11] found a kind of Pseudo-Collision with complexity 2^{16} for MD5 which consists of the same message with two different sets of initial values.

In 1996, H. Dobbertin [9] presented a free start collision with complexity 2^{34} for MD5 during the rump session of EUROCRYPT'96.

In 2005, Wang et.al[15] found collisions with 2^{39} hash operations for MD5.

In 2013,Xie Tao, Fanbaoliu and Dengguo[10] published an attack that breaks MD5 collision resistance in 2^{18} time. This attack runs in less than a second on regular computer.

SHA-1 Hash Function: Secure Hash Algorithm (SHA-1)[4] is based on MD4 , was proposed by the U.S. National Institute for Standards & Technology (NIST) in 1995 for certain U.S federal government applications. The SHA-1 produces a single output of 160-bit from an arbitrary length input message. The input message is composed of multiple blocks each of 512 bits. Each message block is represented as a sequence of sixteen 32-bit words. Following changes are made to obtain SHA-1.

i) Hash value is 160 bits and five (instead of four) 32-bit chaining variables are used.

ii) Compression Function has 4 Rounds. Each round has 20 steps instead of 16.

iii) SHA-1uses 4 non-zero additive constants, whereas MD4 used three constants only two of which were non-zero.

iv) Rotate Left (Circular Left Shift) operation is used. SHA-1 is the most widely used of the existing SHA hash function and is employed in several widely used application and protocols.

Attacks on SHA-1

In 2005, Bihamet. al[9] published a theoretical attack on a reduced version of SHA-1(58 out of 80 rounds) which finds collision with a computational effort of 2^{75} operations (fewer than 2^{80} operations).

In 2005, Wang et. al[12]published an improvement on the SHA-1 attack at the CRYPTO 2005 rump session , lowering the complexity required for finding a collision in SHA-1 to 2^{69}.

In 2010 Marc Steven [8] presents an identical prefix collision attack on SHA-1 with complexities equivalent to approximately 2^{61}(theoretical).

SHA-2 Hash Function: SHA-2[2,3,5] is a set of Cryptographic Hash Function (SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and SHA-512/256) designed by the U.S. National Security Agency (NSA). Significant numbers of changes are made to its predecessor SHA-1. SHA-2 currently consists of a set of six hash functions with digests that are 224, 256, 384 or 512 bits. For SHA-224 and SHA-256, each message block has 512 bits, which are represented as a sequence of 32-bit words. For SHA-384 and SHA-512, each message block has 1024 bits, which are represented as a sequence of 64-bit words. SHA-224 and SHA-256 operate on 32-bit words and SHA-384 and SHA-512 operate on 64-bit words. SHA-256 and SHA-512 are novel hash functions which use different shift amounts and additive constants, but their structures are otherwise virtually identical, differing only in number of rounds. SHA-224 and SHA-384 are simply the truncated versions ofSHA-256 and SHA-512 respectively. SHA-512/224 and SHA-512/256 are also truncated version of SHA-512 but the initial values are generated using the method described in FIPS PUB 180-4[3].

Attacks on SHA-2

The best attack so far on SHA-2 was published in 2011 by Mario Lamberger and Florian Mendel[6] is Pseudo Collision attack against up to 46 rounds of SHA-256.

2.2 Comparative Analysis of Existing Hash Functions (MD5, SHA-1 and SHA-2)

The Comparative analysis of existing hash functions on the basis of different parameters is shown in table1. [3, 7, 8, 9, 10, 13]

Table 1 Comparison between MD5, SHA-1 and SHA-2

Factors	MD5	SHA-1	SHA-2					
			SHA-224	SHA-256	SHA-384	SHA-512	SHA-512/224	SHA-512/256
Max Message Size in bits	2^{64}-1	2^{64}-1	2^{64}-1	2^{64}-1	2^{128}-1	2^{128}-1	2^{128}-1	2^{128}-1
Block Size in bits	512	512	512	512	1024	1024	1024	1024
Message Digest Size in bits	128	160	224	256	384	512	224	256
Word Size in bits	32	32	32	32	64	64	64	64
Rounds	64	80	64	64	64	80	80	80
Collision Resistance Strength in bits	< 64	< 80	112	128	192	256	112	128
Pre-image Resistance Strength in bits	128	160	224	256	384	512	224	256
Second Pre-image Resistance Strength in bits	128	105-160	201-224	201-256	384	394-512	224	256
Speed	Fast	Slow	Slow	Slow	Slow	Slow	Slow	Slow
Collision Found	Yes	Yes(theoretical 2^{61})	No	No	No	No	No	No

3. PROPOSED HASH ALGORITHM

The recent attacks on MD5and SHA-1 reviewed above shows that MD5 and SHA-1 are not collision resistant. These attacks have enforced research in designing of new hash algorithms and cryptanalysis of existing ones. This paper presents the design of new cryptographic hash algorithm which serves the requirement of both the message integrity and source authentication. The proposed algorithm consists of very simple steps and would be secure enough, therefore offers simplicity, security as well as speed.

The proposed algorithm produces 192-bit hash code (digest) from an arbitrary length input and uses a shared secret key of 192-bit (KEY). The sender and receiver have to share secret key of 192-bit (KEY) prior to communication.

Step 1 Key Generation

This algorithm would generate 78 words each of 32-bit from 192-bit shared secret key (KEY). These 78 words are used to form 13 sub keys (Key0 to Key13) , each of 6 words. The process of word formation is as follows:

1. The first six words (W0, W1, W_2, W_3, W4 and W5) are made from the 192-bit key (KEY). The key (KEY) is thought of as an array of 24 bytes (K_0 to K_{23}). The first six bytes (K_0 to K_5) become W0; the next six bytes (K_6 to K_{11}) become W1 and so on. These six words (W0 to W5) become sub key Key0.

2. The rest of the words (W_i, for i=6 to 77) are made as follows

- if (i mod 6≠ 0) then $W_i = W_{i-1} \oplus W_{i-6}$
- if (i mod 6= 0) then $W_i = T_i \oplus W_{i-6}$.

Where T_i is a temporary word obtained as $T_i = W_{i-2} \oplus W_{i-4}$

Step 2 Hash Generation

1. The Plaintext message M is divided into 192-bit blocks M1, M2, M3…....Mn

2. Padding of 0's and 1's should be done to make M a multiple of 192. Padding should be done even if M is already multiple of 192 .

3. Compression Function F

For each block M_i where $1 \leq i \leq N$, Compute

3.1 The 192-bit block is divided into 6 words each of 32-bits

3.2 Copy 6 words into 6 chaining variables W1' to W6' and apply Subkey Key0. This is Preround.

3.3 For each round 1 to 12, Compute

a. W1'=W1'*W6'∧W2' mod 2^{32}
b. W2'=W2'+ShL_{19}(W3')mod 2^{32}
c. W3'=W3'*$RotShft_{24\text{-}21\text{-}7}$ (W2') mod 2^{32}
d. W4'=W4'+ShL_{24}(W5') mod2∧32
e. W5'=W5'*$RotShft_{19\text{-}8\text{-}24}$ (W4') mod 2^{32}
f. W6'=W1'*W6'∧W4' mod 2^{32}

Where $\mathbf{RotShft_{i\text{-}m\text{-}n}(x): RotR_i(x) \oplus RotR_m(x) \oplus ShL_n(x)}$

ShL_i(x): Shift left of argument x by i bits

$RotR_i$(x): Right rotation of argument x by i bits

***: multiplication**

∧: Bitwise AND

+: addition modulo 2^{32}

3.4 Apply Sub keys (1 to 12) to rounds.

4. Calculate Hi= W1'||W3'||W5'||W2'||W4'||W6'

5. S1= H1⊞ H2⊞ KEY mod 2^{192}

6. For i= 2 to N-1

TSH = S_{i-1} ⊞ H_{i+1} mod 2^{192}

TSH is the final digest of 192 bit. Here the symbol || denotes concatenation and ⊞ denotes addition modulo 2^{192}

3.1 Flowchart of Proposed Hash Algorithm Construction Scheme

Figure 1 shows the construction scheme of proposed hash algorithm, this construction scheme is a variant of the famous Merkle-Damgard construction[16,17]

The input message of arbitrary length is divided into N blocks each of 192-bit. Compression Function F is applied to each block M_i, which results in the intermediate digest H_i. An addition modulo 2^{192} operation is applied to first two digests H1 and H2. The result of this operation is then mixes with shared secret key (KEY) by again applying addition modulo operation. This results S1. S1mixes with third intermediate digest H3. This would result S2 and so on. Finally Sn-2 mixes with n^{th} intermediate digest by applying addition modulo operation. The resulting digest TSH is the final digest of 192-bit

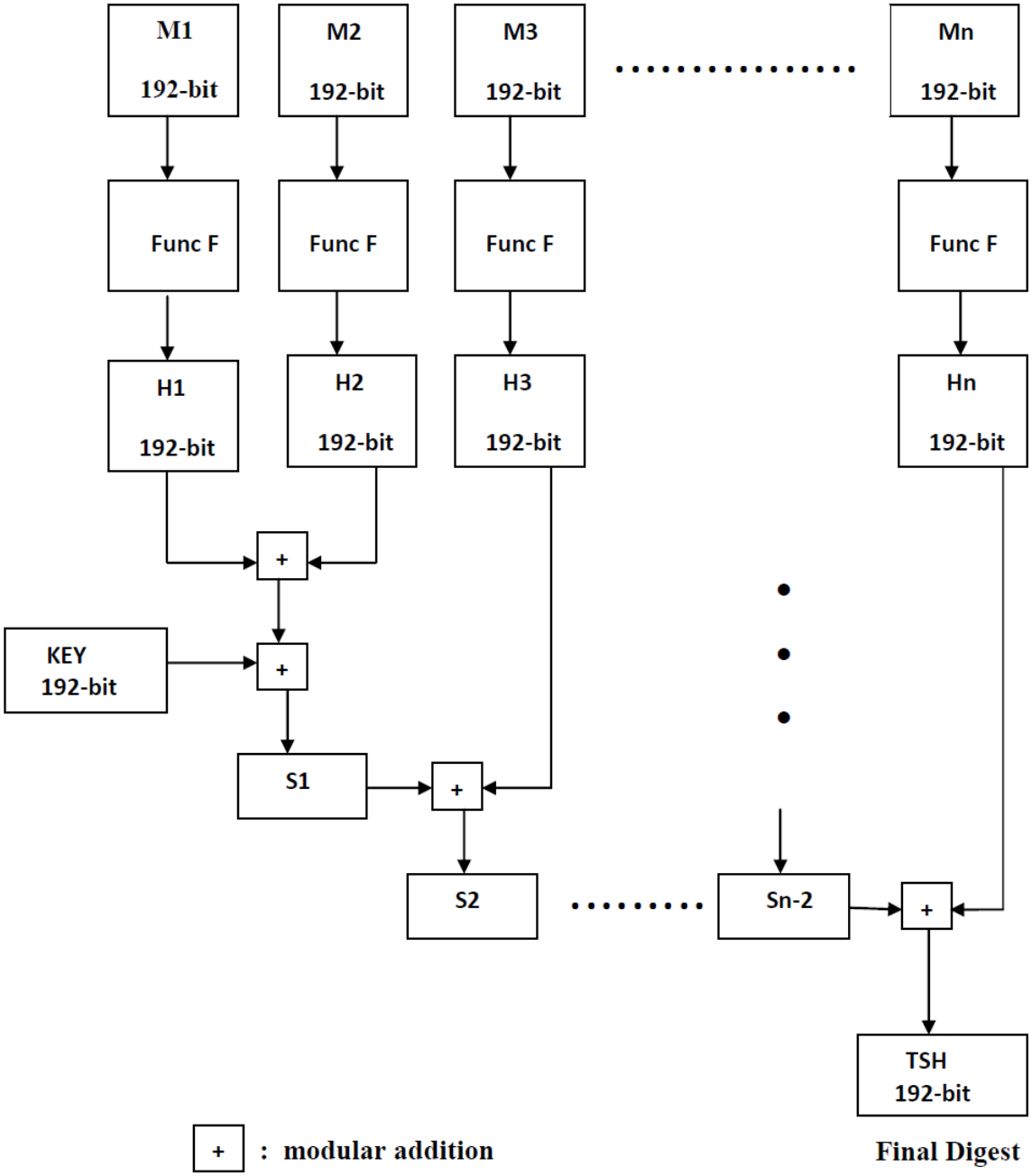
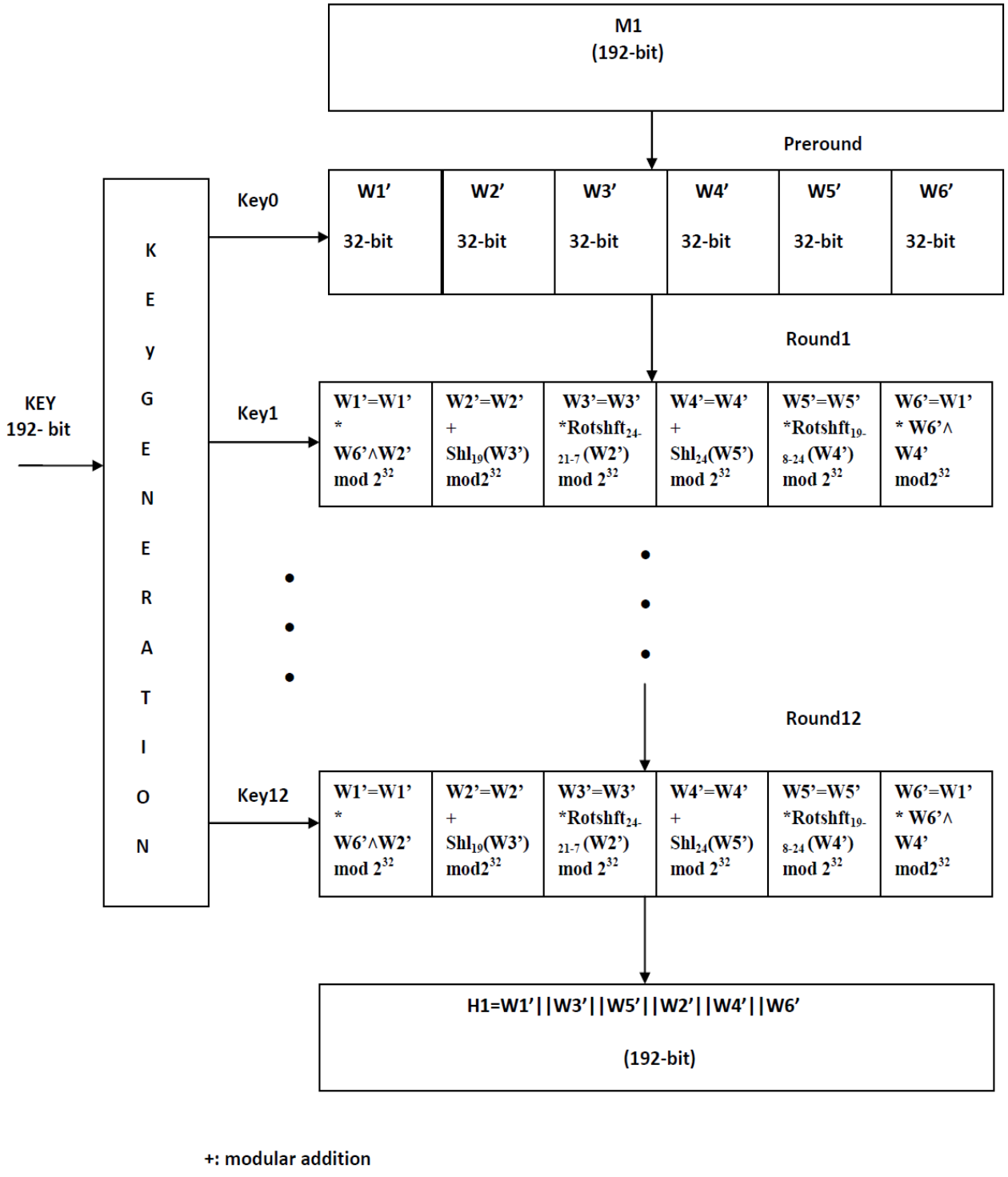


Figure 1. Proposed Hash Algorithm Construction Scheme

3.2. Flowchart of Compression Function F

The Compression Function F for block M1, of proposed algorithm is shown in the Figure 2. The 192-bit block M1 is divided into six words W1'to W6' each of 32 bit. It consists of total of 13 rounds (preround and rest of 12 rounds).Except preround, in each round Bitwise AND, multiplication, RotShft ,ShL and Addition modulo operations are used. The sub keys each of 192-bit, generated during Key Generation process are applied to each round (Key0 to Key12). The compression Function F of proposed hash algorithm would provide strong avalanche effect, as each step takes input from the previous step, which means change in a single bit will leads to change in most of the bits .



.

Figure 2 Compression Function F of Proposed Hash Algorithm

3.3 Security Aspects of Proposed Hash Algorithm

Cryptographic complexity analysis of Hash Algorithms is based on the size of output, they produced. Proposed Hash Algorithm produces digest of 192-bit(greater than MD5 and SHA-1).The proposed algorithm would be secure enough since the operations used in compression function are non invertible and non linear. The proposed hash algorithm would provide strong avalanche effect, as each step takes input from the previous step, which means change in a single bit will lead to change in most of the bits. The algorithm uses the Key during hashing, so any intruder who does not know key, cannot forge the hash code.

4. RESULTS

The proposed algorithm is simulated on Microsoft Visual Studio 2008 platform with Intel based CPU CORE™ i3 2.30GHZ processor with 2GB RAM The table 2 shows the simulation results for sample text data. It is found that the proposed hash algorithm produces strong avalanche effect as the digests produced for slightly different sample data (Hello and Helloo) are significantly different. It is noted that the secret key which is of 192-bit is 24 characters long. Each character represents ASCII 8-bit, so 24×8=192.

The proposed hash algorithm is tested on number of inputs, where input is fixed to a definite size but different secret key is used. In such a case, the execution time is almost same as shown graphically in Figure 3. It is also tested for different input size sample data (10KB to 100 KB) as shown in Figure 4, and It is found that average execution time is proportional to input size. Figure 5 shows that the proposed hash algorithm takes very less time as compared to other hash algorithms since it consists of very simple steps. The digest of message is calculated using the proposed Hash Algorithm is shown by the snapshot of Figure 6.

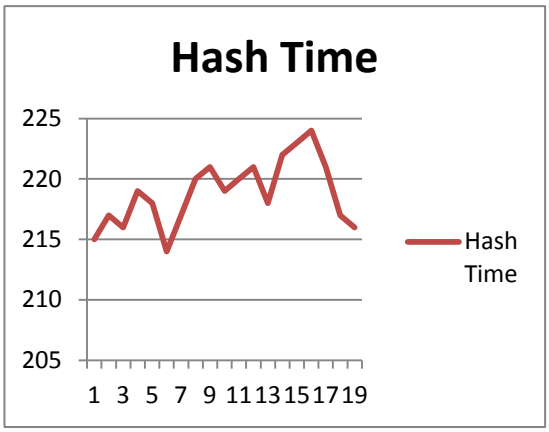


Figure 3 Execution time taken (in ms) by proposed hash algorithm for different input test data of same size but different secret key

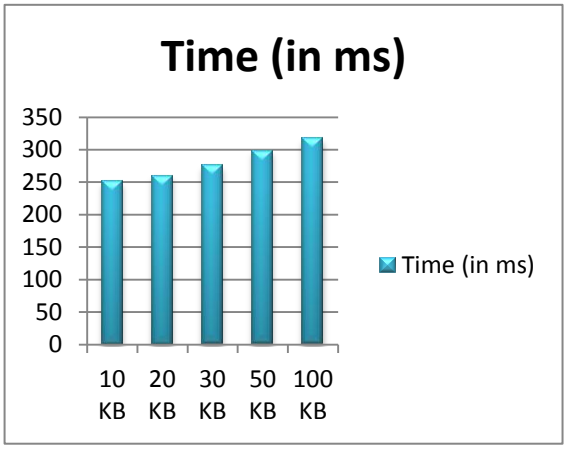


Figure 4 Average Execution Time (in ms) for inputs of different size

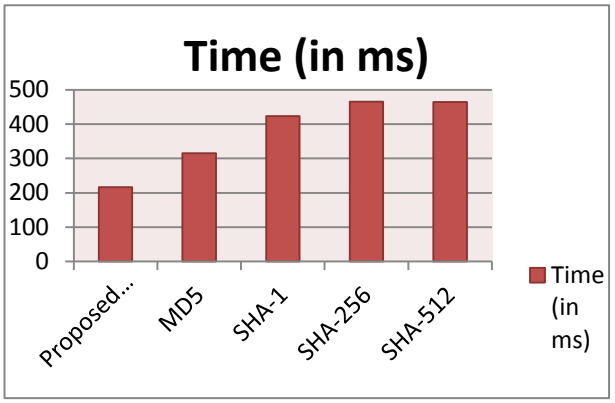


Figure 5 Comparison of execution time taken by few existing hash functions and proposed design

Table 2 Simulation Results for Text data

Sample Data	Secret Key	Message Digest Using Proposed Hash Algorithm
" "	abcdefghijklmnopqrstuvwx	373590b3 d9994703 f033ef07 746cd1c8 1f793534 72f7a942
Hello	abcdefgh1234ijklmnop5678	f01172ee df1c4b79 405606e4 4dc1b95c 469f43aa ccb49e93
Helloo	@#$9876abcdefgh*&123	80ffc87f 208f6e21 b09e4e31 c6511ebb c37dae7b 709b894f
Ram is an extraordinary child.	123456781234567812345678	06e1842b c8f1a013 8001ddcb f3b45b77 9389a64b 08a2bce5
Dear Mom. Please transfer 5000$ to my account as soon as possible. I have to pay my semester fee.	%^#$45678pqrdefgh90123456	b9183e68 cf67beff 62a4933f 29e7a14e 8f3e614c bd9d81e0

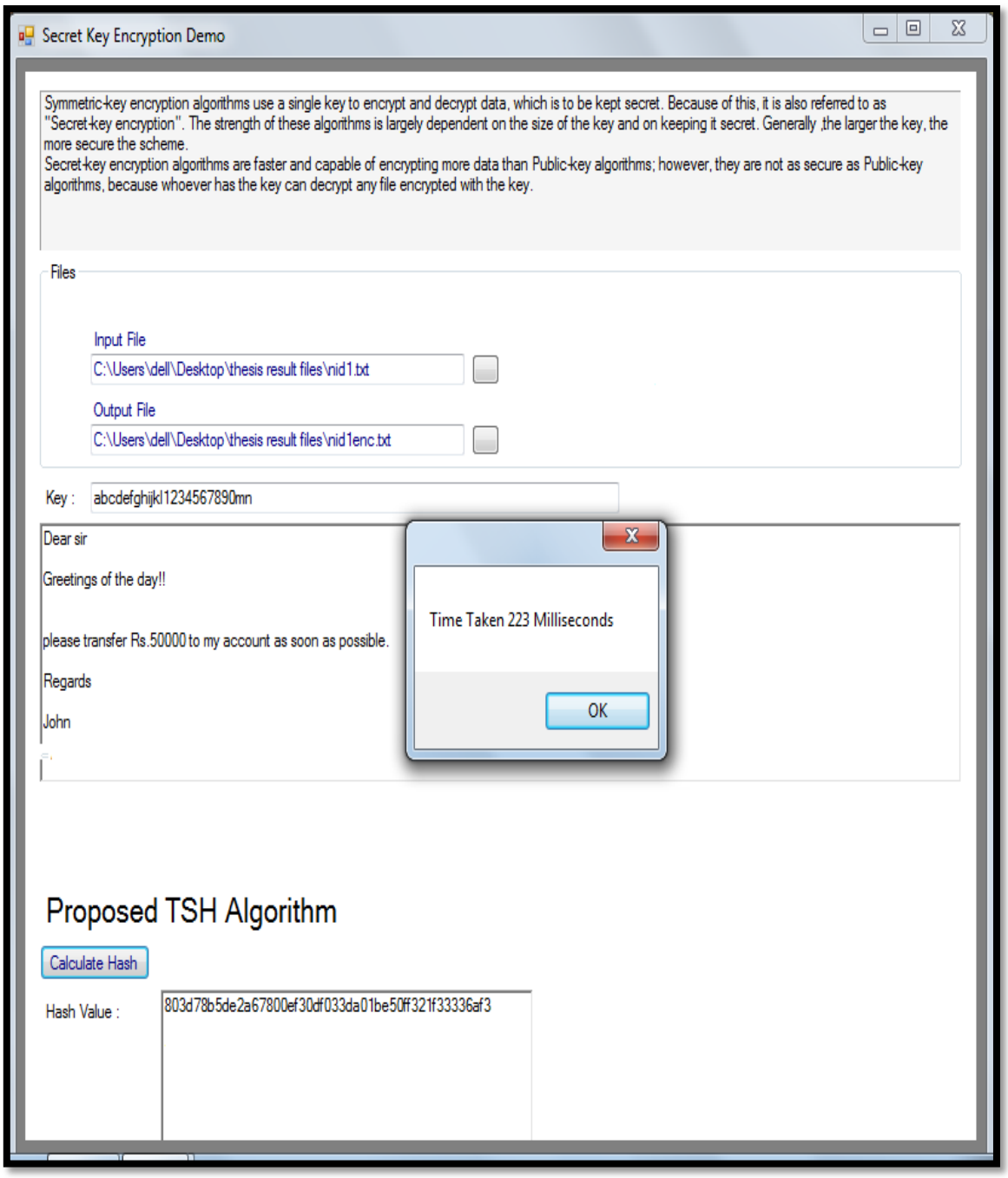


Figure 6 Snapshot for Calculation of hash value of a message using proposed hash algorithm

5. CONCLUSION

This paper focuses on general observation of Hash Algorithms. It has been observed that SHA-512 is better than all other algorithms reviewed in this paper. SHA-512 is more secure in terms of pre-image, second pre-image and collision attacks, but it is slow due to high number of rounds. MD5 is fast but proven inadequate, as now it no longer remains collision resistant. Security of SHA-1 is also questionable. So in this paper a new hash algorithm is proposed, which outputs digest of 192-bit and consists of very simple steps, therefore would have lesser overhead and complexity. The most important feature of this algorithm is that it uses a key as an ingredient to function for calculating the digest, thus providing source authentication as well as message integrity. Based on the simulation results it has been analysed that the proposed algorithm produces strong avalanche effect and takes very less time as compared to other hash algorithms. Thus proposed hash algorithm offers simplicity, security, lesser overhead and complexity and serves the requirement of both the message integrity as well as source authentication.

6. REFERENCES

[1] Saif Al-Kuwari, James H. DavenPort and Rusell J. Bradford. Cryptographic Hash Functions: Recent Design Trends and Security Notions, IACR Cryptology ePrint Archive 2011/565, 2011.

[2] FIPS 180-2.Secure Hash Standard (SHS), National Institute of Standards and Technology, Aug 2002. Replaced by[5]

[3] FIPS 180-4.Secure Hash Standard (SHS) , National Institute of Standards and Technology, March 2012

[4] FIPS 180-1.Secure Hash Standard (SHS), National Institute of Standards and Technology, Apr1995.Replaced by[2]

[5] FIPS 180-3.Secure Hash Standard (SHS), National Institute of Standards and Technology, Oct.2008. Replaced by[3]

[6] Mario Lambergerand Florian Mendel. Higher-Order Differential Attack on Reduced SHA-256, *IACR Cryptology ePrint Archive*. 2011/37,2011

[7] NIST Special Publication-800-107. Recommendation for Applications Using Approved Hash Algorithms, August2012

[8] Marc Stevens. New collision attacks on SHA-1 based on optimal joint local-collision analysis ,Advances in Cryptology – EUROCRYPT 2013, Lecture Notes in Computer Science Volume 7881, 2013, pp 245-261, 2013

[9] IlyaMironov. Hash functions: Theory, attacks and applications, Microsoft Research, Silicon Valley Campus, Nov 14 2005.

[10] Tao Xie, Fanbao Liu, DengguoFeng, "Fast Collision Attack on MD5", IACR, Cryptology ePrint Archive 2013/170 (2013)

[11] B. den Boer, A. Bosselaers. Collisions for the compression function of MD5 , Advances in Cryptology, Eurocrypt'93.1993

[12] Xiaoyun Wang, Yiqun Lisa Yin, and HongboYu. Finding Collisions in the Full SHA-1, IACR, Crypto 2005, LNCS 3621, pp. 17–36 ,2005.

[13] Ronald L.Rivest. The MD5 Message-Digest Algorithm ,Internet Request for comments, April 1992, RFC 1321.

[14] Ronald L. Rivest. The MD4 Message-Digest Algorithm, Internet Request for Comments, October 1990, RFC 1320.

[15] Xiaoyun Wang, Hongbo Yu. How to break MD5 and other Hash Functions, IACR, Eurocrypt 2005, LNCS 3494, 2005

[16] Ivan Damgard. A Design Principle for Hash Functions , In Crypto '89, volume 435 of LNCS, pages 416-427,Springer-Verlag, 1989.

[17] Ralph Merkle. One Way Hash Functions and DES, In Crypto '89, volume 435 of LNCS, pages 428-446. Springer-Verlag, 1989