# A Comparative Study of SHA-256 and the Proposed SHA-288 Hash Model: Algorithmic Design and Analysis

Deepika Reddy, Vishnu Prasad, Akhilesh Rao, Priyanka Menon,
Sahana Murthy & Kavitha Krishnan*ÿIndian Institute of Science (IISc), Bangalore, Karnataka, India

## ABSTRACT

Cryptography is the art of providing security to the message .It gives freedom to the user from hacking of the original message. The here are multiple techniques to provide cryptographic security to the message but this paper focus on the SHA-256 hash function and its extended proposed model SHA-288 for better security. The basic structure of maximum hash functions is based on the Merkle- Damgard construction. Most of the hash functions are used for information security purposes such as digital signature, password authentication; message authentication etc.The hash algorithms perform security checks over plain text by converting plain text into cipher text called message digest or checksums. The paper explains about the cryptanalysis and design of SHA-2 Family as MD5 and SHA-1 going to be outdated after few months. The new model based on construction of SHA-256 has also been highlight for providing strength to the security domain.

**KEYWORDS:** SHA-256/288, Merkle-Damgard construction, Digital Signature, Digital India etc.

## INTRODUCTION

Most of the Cryptographic hash functions are under attacks from last few years. Many types of attacks have been observed on the hash algorithms. Cryptographic functions are one way compression function. Most popular hash functions are producing 128 bits (MD-4, MD-5) and 160 bits (SHA-1) message digest which are no more secure, therefore security community has decided to replace it with SHA-1 variants. Given a hash value, it should require work equivalent to about $2^l$ hash computations any message of length $l$ that hashes to that value. Finding any two messages which hash to the same value should require work equivalent to about $2^{l/2}$ hash computations.

Many attacks have been reported, Out of all attacks, the general attacks are

a) Generic attacks**:** It applies to Merkle-Damgard construction where for p bit hash message digest ,the order more than $2^{p/2}$ are feasible to find attacks on the messages The example of generic attacks are long-message 2nd pre-image attacks [8,14], Joux multicollisions [1],and herding attacks [9].

b) Cryptanalysis attacks**:** This attack is applicable to compression function consisting of Merkle-Damgard construction with multi block collisions on MD-5 and SHA-1 [18, 19].

## PROPERTIES OF HASH FUNCTIONS

Given a function$f: D \to R$, then we say that a hash function $f$ is

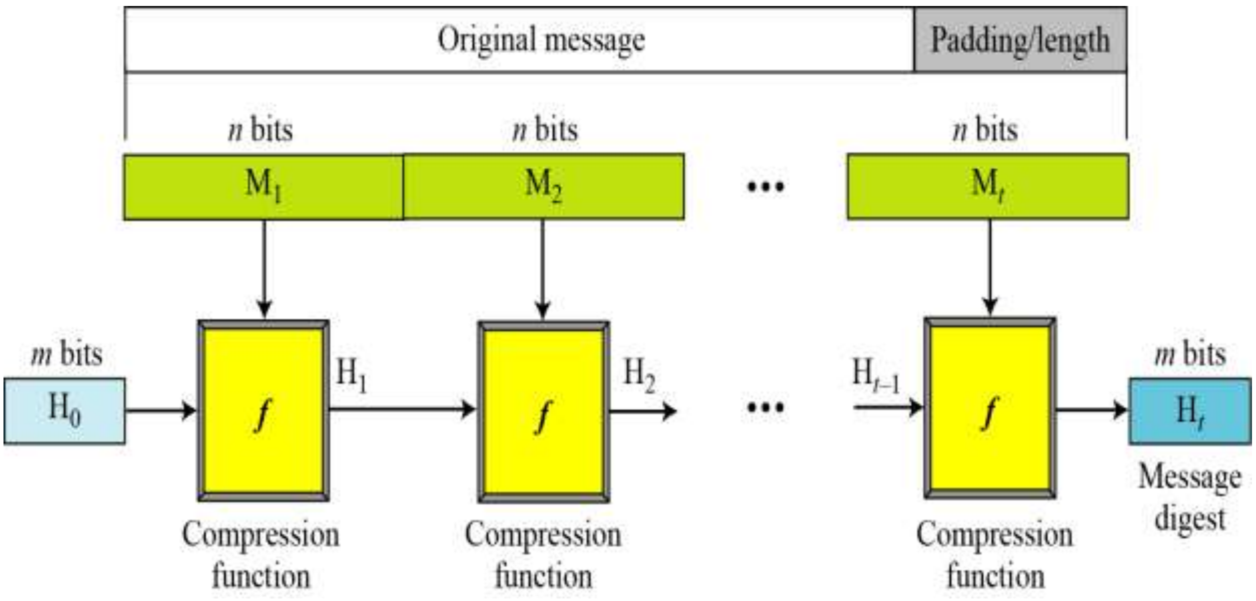
a) Pre-image resistant: For any input $x \in D \ and \ y \in R$, computationally, it is impossible to find a value $x \in D$ such that $f(x) = y$

b) Second pre-image resistant: For any given input $x \in D$ it is very much hard to find a value $x' \in D, x' \neq x$ and $f(x') = f(x)$

c) Collision resistant: It is computationally very much hard to find two distinct values $x', x \in D$ , $f(x') = f(x)$

## MERKLE-DAMGARD CONSTRUCTION-BASIC HASH FUNCTION STRUCTURE

The Hash functions are based on compression functions that is iterated based on Merkle-Damgard Construction[8,13].The hash functions like MD5[1] and SHA-1[1] were used for the security domain but now it has been attacked and going to be obsolete. Both MD-5 and SHA-1 has been attacked by Wang et. al [7, 17] therefore,this paper explains about the ways of getting better security through SHA-256.



## SHA-256 ALGORITHMIC ANALYSIS

SHA-256 provides best possible 128 bits security to the communication system. The message digest of SHA-256 are of 256 bits which has good strength. This is a keyless hash function. The algorithmic analysis of SHA-256 has been explored below.

Basic operations of SHA-256 are as follows:

- Boolean operations AND, XOR and OR, denoted by $\wedge$ ,$\oplus$ and $\vee$ respectively.
- Bitwise complement, denoted by $\neg$
- Integer addition modulo$2^{32}$, denoted by A + B.
- $R^n$ -Right Shift by n bits
- $S^n$ -Right Rotation by n bits

a) The message is divided into blocks of 512 bits which is further divided into 16 blocks of 32 bits each. Each block undergoes 64 rounds of round functions. Each 32bits word goes through the different operations as described below.

b) The initial hash value $H^{(0)}$ is calculated by getting fractional part of square root of first 8 prime numbers. The fractional part is converted into binary then into hexadecimal forms.

$H_1^0$= 6a09e667; $H_2^0$=bb67ae85; $H_3^0$=3c6ef372; $H_4^0$= a54ff53a; $H_5^0$= 510e527f; $H_6^0$ = 9b05688c;$H_7^0$= = 1f83d9ab; $H_8^0$= = 5be0cd19

c) Padding the message: The length of message should be multiple of 512 bits. If the total length of message is $\boldsymbol{l}$ then "1" is appended at the end of the message. The condition$l + 1 + k \equiv 448 mod 512$ should be satisfied.

d) Now parse the message into N 512-bit blocks $M^1, M^2, M^3, M^4, \ldots.. M^N$ ,the first 32 bits of message block $i$ are denoted $M_0^i, M_1^i, M_2^i \ldots. M_{15}^i$ .The algorithm conventions are based on big-endian style.

e) The main loop will be followed as below:

For $i$=1 to $N$ ($N$ is number of blocks in the padded message) {

Initialize register $a, b, c, d, e, f, g, h$ with the $(i-1)^{st}$ with intermediate hash value while running the loop

$a \leftarrow H_1^{(i-1)}$;$b \leftarrow H_2^{(i-1)}$;$c \leftarrow H_3^{(i-1)}$;$d \leftarrow H_4^{(i-1)}$;

$e \leftarrow H_5^{(i-1)}$;$f \leftarrow H_6^{(i-1)}$;$g \leftarrow H_7^{(i-1)}$;$h \leftarrow H_8^{(i-1)}$

Now, after running the loop, the compression function is applied to updated the register $a, b, c, d, e, f, g, h$

For $j = 0 \; to \; 63$ {

Compute

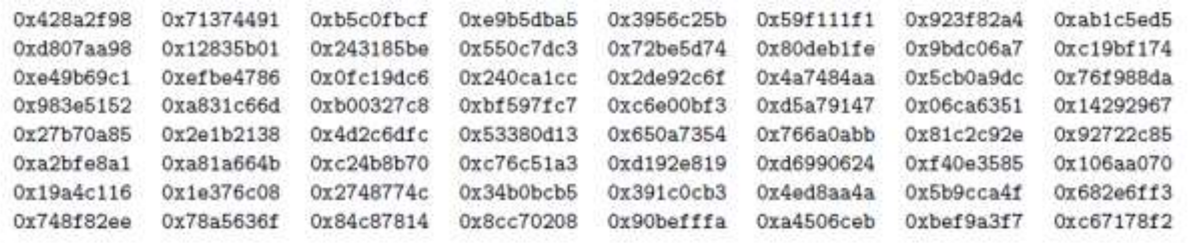
Ch $(e, f, g)$; $Maj(e, f, g)$; $\sum_0(a)$ ; $\sum_1(e)$ and $W_j$

$T_1 \leftarrow h + \sum_1 e + Ch(e, f, g) + K_j + W_j$ (Here + is mod $2^{32}$ Addition)

$T_2 = \sum_0 a + Maj(a, b, c)$
$h \leftarrow g;\ g \leftarrow f;$
$f \leftarrow e;$
$e \leftarrow d + T_1;$
$d \leftarrow c;$
$c \leftarrow b;$
$b \leftarrow a;$
$a \leftarrow T_1 + T_2;\}$
Compute the $i^{th}$ intermediate hash value $H^{(i)}$
$H_1^{(i)} \leftarrow a + H_1^{(i-1)}; H_2^{(i)} \leftarrow a + H_2^{(i-1)}; H_3^{(i)} \leftarrow a + H_3^{(i-1)}; H_4^{(i)} \leftarrow a + H_4^{(i-1)};$
$H_5^{(i)} \leftarrow a + H_5^{(i-1)} \quad ; \ H_6^{(i)} \leftarrow a + H_6^{(i-1)}; H_7^{(i)} \leftarrow a + H_7^{(i-1)}; H_8^{(i)} \leftarrow a + H_8^{(i-1)}; \}$
Finally compute the hash function
$H^{(N)}{=}H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)} \| H_8^{(N)}$
The compression functions of SHA-256 are as follows

$$Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$Maj(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum\nolimits_0 (x) = S^2\,(x) \oplus S^{13}\,(x) \oplus S^{22}\,(x)$$
$$\sum\nolimits_1 (x) = S^6\,(x) \oplus S^{11}\,(x) \oplus S^{25}\,(x)$$
$$\sigma_0(x) = S^7\,(x) \oplus S^{18}\,(x) \oplus R^3\,(x)$$
$$\sigma_1(x) = S^{17}\,(x) \oplus S^{19}\,(x) \oplus R^{10}\,(x)$$

The message block $W_0, W_1, W_2, W_3, W_4, \ldots \ldots \ldots W_{62}, W_{63}$ are computed as below
$W_j = M_j^i \ for \ j = 0,1, \ldots.15, and$
$For \ j = 16 \ to \ 63$
$\{\ W_J \leftarrow \sigma_1\big(W_{j-2}\big) + W_{j-7} + \sigma_0\,\big(W_{j-15}\big) + W_{j-16}\}$
There are sixty four constants $K_0, \ldots \ldots \ldots .. K_{64}$ words produced by cube roots of fractional part of first sixty four prime numbers.



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x428a2f98 | 0x71374491 | 0xb5c0fbcf | 0xe9b5dba5 | 0x3956c25b | 0x59f111f1 | 0x923f82a4 | 0xab1c5ed5 |
| 0xd807aa98 | 0x12835b01 | 0x243185be | 0x550c7dc3 | 0x72be5d74 | 0x80deb1fe | 0x9bdc06a7 | 0xc19bf174 |
| 0xe49b69c1 | 0xefbe4786 | 0x0fc19dc6 | 0x240ca1cc | 0x2de92c6f | 0x4a7484aa | 0x5cb0a9dc | 0x76f988da |
| 0x983e5152 | 0xa831c66d | 0xb00327c8 | 0xbf597fc7 | 0xc6e00bf3 | 0xd5a79147 | 0x06ca6351 | 0x14292967 |
| 0x27b70a85 | 0x2e1b2138 | 0x4d2c6dfc | 0x53380d13 | 0x650a7354 | 0x766a0abb | 0x81c2c92e | 0x92722c85 |
| 0xa2bfe8a1 | 0xa81a664b | 0xc24b8b70 | 0xc76c51a3 | 0xd192e819 | 0xd6990624 | 0xf40e3585 | 0x106aa070 |
| 0x19a4c116 | 0x1e376c08 | 0x2748774c | 0x34b0bcb5 | 0x391c0cb3 | 0x4ed8aa4a | 0x5b9cca4f | 0x682e6ff3 |
| 0x748f82ee | 0x78a5636f | 0x84c87814 | 0x8cc70208 | 0x90befffa | 0xa4506ceb | 0xbef9a3f7 | 0xc67178f2 |

## EXTENSION OF SHA-256 STRENGTH-PROPOSED MODEL

Now introducing one new initializing variable into the operation such that constant variable $H_9^{(i)} =$ CBBB9D5D(Converting into hexadecimal form of square root of ninth prime number (i.e.23)
Sqrt(23)=4.795831523312719541597438064162693919996707041904129346485309114
Binary equivalent of above number is
100.11001011101110111001110101011101110000010000001011
Now converting into hexadecimal we get CBBB9D5D.
The operation of the hash value in compression functions $i \leftarrow h \ and \ calculating \ \ H_9^{(i)} \leftarrow a + H_9^{(i-1)}$ and then hash output after concatenation

$H^{(N)}{=}H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)} \| H_8^{(N)} \| H_9^{(N)}$
It will be of 288 bits which may be stronger than SHA-256 hash functions.

## CONCLUSION

If we extend the message digest length of SHA-256 from 256 bits to 288 bits as explained above then application will be more secure perfect**.** We are talking about the digital India and cashless economy, but the security concern is the big challenge ahead because we would be using the following security dimension in our daily activities which would not be secure without better algorithms.

- Digital Transaction
- Message authentication

- Software integrity
- One-time Passwords
- Digital signature
- Time stamping
- Certificate revocation management

The Security strength of SHA-256 which is going to be used in coming days are as follows:

| **Algorithm** | Message Size (bits) | Block Size (bits) | Word Size (bits) | Message Digest Size (bits) | Extended Message Length |
|---|---|---|---|---|---|
| **SHA-256** | $< 2^{64}$ | 512 | 32 | 256 | 288 |

## FURTHER SCOPE OF RESEARCH

The SHA-2 and SHA-3 family should be analyzed for better security threat tolerance.
Researchers can find the strength and weakness of SHA-256 and proposed SHA-288 family algorithms.

## REFERENCES

[1] Antoine Joux. Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions. In Matt Franklin, editor, *Advances in Cryptology-CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*,pages 306–316, Santa Barbara, California, USA, August 15–19 2004. Springer.

[2] B.Preneel, V. Rijmen, A.Bosselaers: Recent Developments in the Design of Conventional Cryptographic Algorithms. In State of the Art and Evolution of Computer Security and Industrial Cryptography. LNCS 1528. Springer-Verlag, Berlin Heidelberg New York(1998) pp.106-131.

[3] E.Biham and R.Chen. Near-Collisions of SHA-0,In Advances in Cryptology CRYPTO'2004, LNCS 3152,p..290-305, 2004.

[4] Eli Biham and Orr Dunkelman. A Framework for Iterative Hash Functions-HAIFA. Technical report, August 2006. The paper and slides of this work are available at http://csrc.nist.gov/pki/HashWorkshop/2006/ program_2006.htm. Last access date: 15th of February 2007.

[5] Helena Handschuh and David Naccache. SHACAL, 2001. Available at https://www.cosic.esat.kuleuven.ac.be/nessie/tweaks.html/shacal tweak.pdf. 15. Carlo Harpes, Gerhard Kramer, and James Massey. A generalization of linear cryptanalysis and the applicability of Matsui's Piling-up lemma. In Louis Guillou and Jean-Jacques Quisquater, editors, Advances in Cryp- tology - Proceedings of EUROCRYPT 95, volume 921 of Lecture Notes in Computer Science, pp. 24-38. Springer-Verlag, 1995

[6] I.Damg˚ard. A design principle for hash functions. In G. Brassard, editor, Advances in Cryptology-CRYPTO'89, LNCS 435. Springer-Verlag, 1990.

[7] Joan Daemen, Michael Peeters, and Gilles Van Assche. RadioGatun, a Belt-and-Mill Hash Function. Technical report, August 2006. The paper and slides of this work are available at http://csrc.nist.gov/pki/HashWorkshop/2006/program_2006.htm. Last access date: 15th of February 2007.

[8] John Kelsey and Bruce Schneier. Second Preimages on n-bit Hash Functions for Much Less than 2ˆn Work. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in ComputerScience*, pages 474–490. Springer, 2005.

[9] John Kelsey and Tadayoshi Kohno. Herding Hash Functions and the Nostradamus Attack. In Serge Vaudenay,editor, *Advanes in Cryptology-EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 183–200. Springer, 2006.

[10] Praveen Gauravaram, William Millan, Ed Dawson, Matt Henricksen, Juanma Gonzalez Nieto, and Kapali Viswanathan. Constructing Secure Hash Functions by Enhancing Merkle-Damg˚ard Construction (full version).

[11] Praveen Gauravaram,William Millan, Ed Dawson, and Kapali Viswanathan. Constructing Secure Hash Functions by Enhancing Merkle-Damg˚ard Construction. In *Australasian Conference on Information Security and Privacy(ACISP)*, volume 4058 of *Lecture Notes in Computer Science*, pages 407–420, 2006.

[12] Praveen Gauravaram. *Cryptographic Hash Functions: Cryptanalysis, Design and Applications*. PhD thesis,Information Security Institute, Queensland University of Technogy, June 2007.

[13] R.C.Merkle, One Way Hash Functions and DES, In G. Brassard, editor, Advances in Cryptology-CRYPTO' 89, LNCS 435 Springer-Verlag, pp.428-446, 1990.

[14] Richard Drews Dean. *Formal Aspects of Mobile Code Security*. PhD thesis, Princeton University, 1999.

[15] Ronald Rivest. Abelian Square-free Dithering and Recoding for Iterated Hash Functions. Technical report,October 2005. The paper and slides of this work are available at http://csrc.nist.gov/pki/HashWorkshop/2005/program.htm. Last access date: 15th of February 2007.

[16] Technical Report QUT-ISI-TR-2006-013, Information Security Institute (ISI), Queensland University of Technology (QUT), July 2006. This technical report is available at http://www.isi.qut.edu.au/research/publications/technical/qut-isi-tr-2006-013.pdf. Last access date: 12th of september 2007.

[17] X. Wang, X.Lai, D.Feng and H.Yu., Cryptanalysis of the Hash Functions MD4 and RIPEMD, EUROCRYPT 2005, LNCS 3494, pp.1-18, Springer-Verlag, 2005. 42. X. Wang, H. Yu, How to Break MD5 and Other Hash Functions, EUROCRYPT'2005, Springer-Verlag, LNCS 3494, pp.19-35, 2005.

[18] Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Ronald Cramer, editor,*Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35.Springer, 2005.

[19] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In Victor Shoup, editor,*Advances in Cryptology—CRYPTO '05*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005, 14–18 August 2005.