International Journal of Scientific Research in
Network Security and Communication
Vol.**13**, Issue.**1**, pp.**08-11**, February **2025**
ISSN: 2321-3256 (Online)
Available online at: www.ijsrnsc.org

Research Article

Enhancing the Performance of Cryptographic Hash Function Using 2080 Bits Proposed Secure Hash Algorithm 160

Bhagvant Ram Ambedkar[1]

[1]Dept. of Computer Science and Information Technology, M. J. P. Rohilkhand University, Bareilly, Uttar Pradesh, India

**Corresponding Author:* ✉

Received: 18/Dec/2024, Accepted: 20/Jan/2025, Published: 28/Feb/2025| **DOI**: https://doi.org/10.26438/ijsrnsc.v13i1.264

Abstract— an on-way hash code or message authentication code is generated using the cryptographic hash functions. It used to be password storage, electronic data integrity, and check verification. Cryptographic hashing algorithms, which employ beginning value and key constant to boost algorithm complexity, have been proposed by a number of academics. It is well known that they have a very high temporal complexity due to the quantity of steps and memory space needed to store the beginning value and key constants. Consequently, we are improving the cryptographic hash function's performance by using 2080 bits as a block of the input message and avoiding the need for the key constant. By doing this, we are generating 160-bit fixed-length hash code, and the amount of time spent on the function proposal will be reduced in comparison to previous hash algorithms. The outcome will be compared using the amount of time in seconds that the cryptographic hash algorithms consumed during computation.

Keywords— Cryptography, Hash Function, Information Security, Key Constant

I. INTRODUCTION

Nowadays, there is a growing and widespread need for electronic data communication via the Internet; everyone wants to communicate data quickly and securely. How to quickly and securely verify e-data during internet communication is a major security concern since the hash function verifies e-data sent over the internet. As a result, there is a lot of room for study into secure hash algorithms (SHA), and numerous academics have designed and assessed SHA's performance. The hash code is a fixed length code of the variable length input message, which is computed by SHA [1]. It serves as a hash code and is employed in information authentication security [2]. For variable-length input messages, hash functions generate fixed-size hash codes [3]. Hash algorithm applications that are efficient and low-power have recently been created for the dynamic field [4]. Developing global enterprises employ this fundamental cryptographic approach to confirm the confidentiality and authenticity of web data [5]. It is extremely challenging to create a coding theory-based electronic data verification system that is both safe and effective [6]. The hash functions provide the protection and privacy of electronic data [7]. One major problem is the pervasive and growing need for social media and safe online information transfer over public networks [8]. Techniques for cryptographic hash functions are employed to ensure sensitive data security and authenticity [9]. The enhancement and implementation of the hash algorithm for secure data communication through the web [10]. By implementing the proposed algorithm we can provide the authentication of message data communication through the insecure channel [11]. The security of electronic data is based on a secure hash algorithm [12]. Cryptographic hash functions provide very important roles such as digital signature, message integrity, and authentication [13]. It is very efficient in case those devices have limited memory [14]. The basic operation of our proposed algorithm uses bitwise logical operation [15]. Protecting smart devices is a big issue because they have limited memory space [16]. We can encrypt the image based on chaos and a secure hash algorithm [17]. Hashing algorithms satisfy security requirements and prove logical and arithmetic operations [18]. To produce a 160-bit hash code that is independent of the starting value, this suggested approach uses 13-step procedures in each round. Reconstructing the original message is therefore extremely challenging because this technique uses 2080 input blocks and 13 function processing steps per round, regardless of the beginning value, which is disclosed to the public [2]. The main advantage of this technique is that the starting value and key constant are stored without the need for buffer RAM.

II. ISSUED WITH CRYPTOGRAPHIC HASH FUNCTION

A. Hah Function

Due to its inability to reconstruct the original message from the hash code, the hash function produces a one-way hash code. Since digital data fluctuates between 0 and 1, it is particularly difficult to conceal and confirm the integrity of secure digital data. These days, hash techniques are used to secure passwords and confirm their authenticity [1]. The property that the function's outputs are uniquely random when applied to a large collection of inputs with padding bits is satisfied by the perfect hashing techniques. As a cryptographic hash function, the blameless hashing techniques are necessary for safety applications.

B. Padding

Padding plays a crucial part in hashing algorithms by converting variable-size input messages into fixed blocks of messages with a fixed size. Increasing the number of padding bits in the hash function will strengthen its security. Thus, we are concentrating on padding the bit and applying it to the techniques we have suggested. In hashing algorithms, padding is crucial; the input message is secured using padding techniques [17]. A problem with hashing algorithms is that they add a lot of zero-bit integers and padding, followed by a single bit [18]. For example:

Input Message = xyz

Require a total number of padding bits (P) = (1920 - 24) mod 2080 = 1896 bits

10000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000

Length of Padding bit = 1896

III. PROPOSED ALGORITHM

The hash function's starting value (H_0) is the main subject of this study. Numerous academics have suggested an existing approach that uses a hash function (H_n) or message digest with a specified, identical beginning value. When multiple input messages are identical, it becomes extremely challenging for us to distinguish between them. Since we obtain the same hash function (Hn) in this instance [1]. For this reason, we are suggesting a PSHA160 to produce a 160-bit secure hash function. The suggested algorithm's secure hash function will be unaffected by the additive constant Kt and the beginning value (IV or H_0). The basic architecture of PSHA 160 is shown in Fig. 1

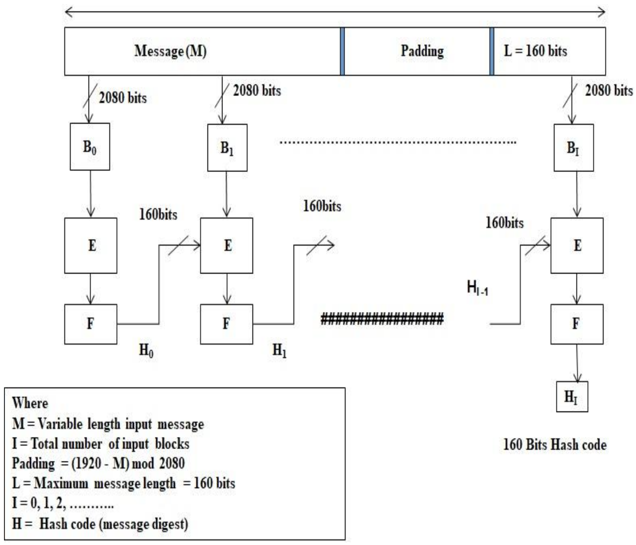

Figure 1. The basic hashing function of PSHA 160

It is particularly helpful for electronic data verification. PSHA160 performs the subsequent actions:

1. Choose any input message (M) of variable length
2. Input messages are separated into blocks
3. Compute the length of padding bits
4. Length of block with padding bit (E)= 2080 bits
5. E = M + P + L = 2080 bits
6. E is separated into thirty-two sub-blocks (E0, E1, ………. E12), and each block size will be 160 bits.
7. Utilizing logical NOR functions (F) in 12 steps and logical Ex-OR functions (F) in 11 steps, create the hash function as illustrated in Fig. 2.

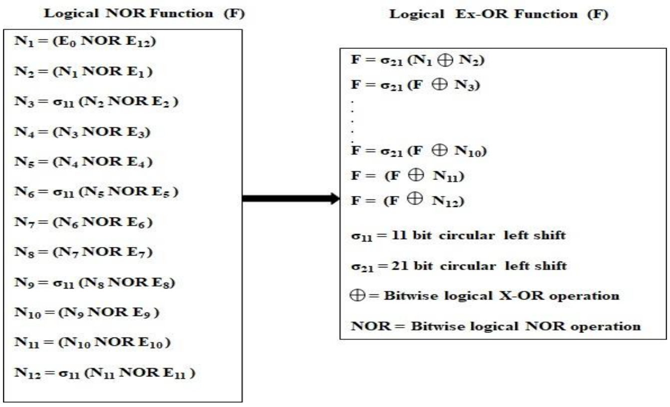

Figure 2. Single-round hash function processing of PSHA-160

IV. EXPERIMENTAL RESULTS

Experimental results of PSHA 160 and hashing algorithms are executed by python-3.9.0 on Windows 10, 64-bit Operating system, 4GB RAM platform, and Intel(R) processor shown in Table 1.

Table 1. Experimental Executed Results of Hashing Algorithms with Variable Length Input Message

Hash Algorithms	Input Message Length in Bits	Elapsed time in the Second
PSHA-160	112	0.00011179999955857056
	152	0.00010250000013911631
	88	0.00015970000004017493
SHA256	112	0.0004322000000058779
	152	0.0003767000000607368
	88	0.0002563000000463944
SHA384	112	0.00033790000003364185
	152	0.00020289999997658015
	88	0.0003958999998303625
SHA224	112	0.00043310000000928994
	152	0.0002565000000913642
	88	0.0003089999997882842
SHA512	112	0.00027039999991984587
	152	0.00010189999989052012
	88	0.00015729999995528487
SHA1	112	0.0003123999999843363
	152	0.0006099000000858723
	88	0.00038560000029974617

V. ANALYSIS OF RESULTS

To avoid the preimage and second preimage attacks, the input message length for a map with an n-bit hash code size will be less than 2^n. Selecting values of x at random and trying each one until a collision happens is known as a brute-force attack. The amount of work required for an n-bit hash value is proportional to 2n, and it attempts, on average, 2^{n-1} values of x to identify one that produces a certain hash value h. [2, 19].

The above security requirements satisfy our proposed algorithm so our proposed algorithm is secure and time efficient because it takes to order one complexity O(1) during all phases of function processing.

Our proposed algorithm satisfies all security requirements:

Preimage Resistance:

M = wxyz

M in bits = 32

PSHA-160 hash code= d60d69d90c143d60d69d90c143d60d1ea14d6943

The input message is the preimage of the hash code.

Collision Resistance:

M = zyxw

M in bits = 32

PSHA-160 hash code = dd25f98a5a44bdd25f98a5a44bdd2583f31cdc4b

As shown in the above-executed example H(wxyz) ≠ H(zyxw), it is collision-resistant.

Second Preimage:

It is quite challenging to locate any two input messages (x, y) that have the hash code H(x) = H(y).

Twelve logical NOR operations used in this study swap maximum zeroes for one and vice versa. Nineteen Exclusive-OR logical operations, twelve round steps, and an 11-bit circular left shift (CLS) four-step operation with bitwise logical NOR and a 21-bit CLS with bitwise logical Exclusive-OR ten-step operation are used for function operations. PSHA-160's executed results with existing hashing algorithms are executed using the built-in Python tools of import hashlib, as indicated in Table 1. The comparative executed results of elapsed time in seconds with variable size input message length are shown in Fig. 4, and the basic parameter comparative analysis of hashing algorithms with PSHA-160 is shown in Fig. 3.

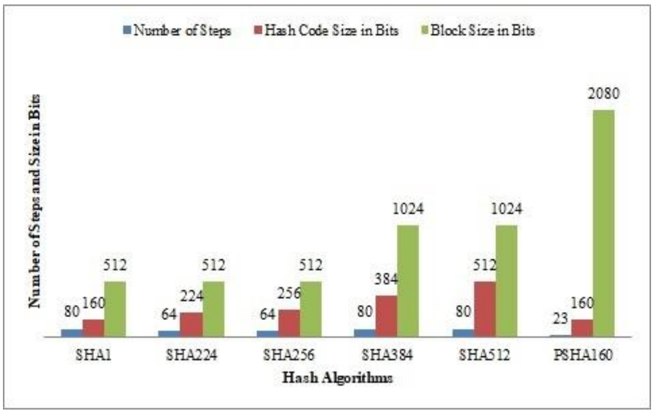

Figure 3. Comparative analysis of Hashing Algorithm with PSHA 160

The elapsed time in seconds consumed by hashing algorithms during computation is shown in Fig. 4. It may varied with computing devices will changed.

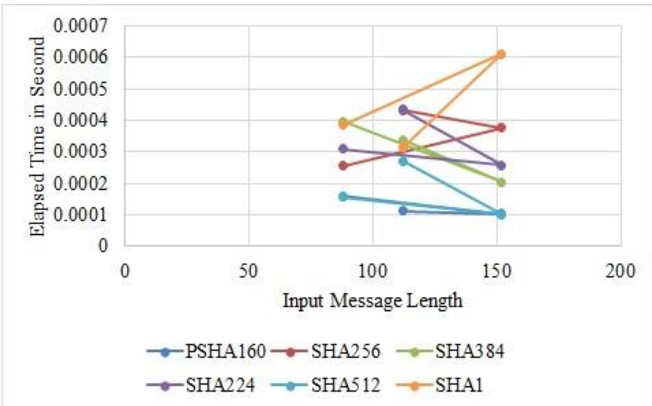

Figure 4. Comparative Expperimental Results of Hashing Algorithms with Elapsed Time in Second

VI. CONCLUSION

A secure hash technique that is independent of key constants employed by cryptographic hash methods was proposed in this study. It is using 2080 bits as the input message to generate a 160-bit secure hash algorithm. Because it uses less elapsed time than other hash algorithms, it improves the cryptographic hash function's performance. Since there is no specified keyword needed and the input block size for function processing is 2080 bits, we can assert that this hash technique is secure. This algorithm's primary drawback is its weak security when input messages contain fewer than three characters. Future research can address this issue, and this algorithm's benefit is that it is extremely challenging to reconstruct the original message from the hash code because there isn't an input variable that is publicly known.

Conflict of Interest

Author has no conflict interest and no has any Funding source.

Acknowledgment

I would like to sincerely thank and be obliged to Professor P. K. Bharti and Dr. Akhtar Husain for their unwavering support, capable direction, and insightful recommendations in writing this paper.

.

REFERENCES

[1] B. R. Ambedkar, P. K. Bharti, Akhtar Husain."Enhancing the Performance of HashFunction Using Autonomous Initial ValueProposed Secure Hash Algorithm 256", *2022IEEE 11th International Conference onCommunication Systems and NetworkTechnologies (CSNT),* **2022.**

[2] B. R. Ambedkar, P. K. Bharti, Akhtar Husain."Design and Analysis of Hash Algorithm UsingAutonomous Initial Value Proposed SecureHash Algorithm64", *2021 IEEE 18th IndiaCouncil International Conference (INDICON),***2021**

[3] S. Mathew and K. P. Jacob, "Performance Evaluation of Popular Hash Functions," *World Academy of Science, Engineering and Technology*, pp.**449–452, 2010.**

[4] X. Zheng, X. Hu, J. Zhang, J. Yang, S. Cai, and X. Xiong, "An Efficient and Low-Power Design of the SM3 Hash Algorithm for IoT," *Electronics, vol. 8, no. 9, Art. no. 9*, Sep. **2019**, doi: 10.3390/electronics8091033.

[5] A. A. Yavuz and M. O. Ozmen, "Ultra Lightweight Multiple-time Digital Signature for the Internet of Things Devices," *IEEE Transactions on Services Computing*, pp.1–1, **2019**, doi: 10.1109/TSC.2019.2928303.

[6] P. Santini, M. Baldi, and F. Chiaraluce, "Cryptanalysis of a One-Time Code-Based Digital Signature Scheme," *in 2019 IEEE International Symposium on Information Theory (ISIT)*, Jul., pp.**2594–2598, 2019.** doi: 10.1109/ISIT.2019.8849244.

[7] A. Mohammed Ali and A. Kadhim Farhan, "A Novel Improvement With an Effective Expansion to Enhance the MD5 Hash Function for Verification of a Secure E-Document," *IEEE Access*, vol.8, pp.**80290–80304**, **2020**, doi: 10.1109/ACCESS.2020.2989050.

[8] M. Samiullah et al., "An Image Encryption Scheme Based on DNA Computing and Multiple Chaotic Systems," *IEEE Access*, vol. 8, pp. 25650–25663, **2020**, doi: 10.1109/ACCESS.2020.2970981.

[9] L. Singh, A. K. Singh, and P. K. Singh, "Secure data hiding techniques: a survey," *Multimed Tools Appl*, vol. 79, no. 23, pp. 15901–15921, Jun. 2020, doi: 10.1007/s11042-018-6407-5.

[10] F. E. De Guzman, B. D. Gerardo, and R. P. Medina, "Implementation of Enhanced Secure Hash Algorithm Towards a Secured Web Portal," *in 2019 IEEE 4th International Conference on* Computer and Communication Systems (ICCCS), Feb., pp.**189–192, 2019.** doi: 10.1109/CCOMS.2019.8821763.

[11] A. Faz Hernández, H. Fujii, D. Aranha, and J. López, "A Secure and Efficient Implementation of the Quotient Digital Signature Algorithm (qDSA)", pp.189, **2017**. doi: 10.1007/978-3-319-71501-8_10.

[12] X. Fei, K. Li, W. Yang, and K. Li, "A secure and efficient file protecting system based on SHA3 and parallel AES," *Parallel Computing*, Vol.**52**, pp.**106–132**, **2016**, doi: 10.1016/j.parco.2016.01.001.

[13] B. Madhuravani and D. S. R. Murthy, "Cryptographic hash functions: SHA family," *International Journal of Innovative Technology and Exploring Engineering (IJITEE*), Vol.**2**, No.**4**, pp.**326–329, 2013.**

[14] K. Ideguchi, T. Owada, and H. Yoshida, "A Study on RAM Requirements of Various SHA-3 Candidates on Low-cost 8-bit CPUs," 260, 2009. Accessed: Nov. 23, **2021**.

[15] V. A. Melnyk and A. Y. Kit, "Basic Operations of Modern Hashing Algorithms," *COMPUTER SCIENCE*, p. 4, **2013.**

[16] W. Liang, S. Xie, J. Long, K.-C. Li, D. Zhang, and K. Li, "A double PUF-based RFID identity authentication protocol in service-centric internet of things environments," *Information Sciences,* Vol.503, pp.**129–147**, Nov. **2019**, doi: 10.1016/j.ins.2019.06.047.

[17] S. Zhu, C. Zhu, and W. Wang, "A new image encryption algorithm based on chaos and secure hash SHA-256," *Entropy*, Vol.20, No.9, p.716, **2018**.

[18] L. Grassi, D. Khovratovich, C. Rechberger, A. Roy, and M. Schofnegger, "Poseidon: A New Hash Function for Zero-Knowledge Proof Systems," *presented at the 30th {USENIX} Security Symposium ({USENIX} Security 21),* 2021, pp. 519–535. Accessed: Nov. 23, **2021.**

[19] A. Kuznetsov, M. Lutsenko, K. Kuznetsova, O. Martyniuk, V. Babenko, and I. Perevozova, "Statistical Testing of Blockchain Hash Algorithms," p. 13, **2020.**

AUTHORS PROFILE

Dr. Bhagwant Ram Ambedkar received the B. Tech. Degree in Electronics Engineering from the Institute of Engineering and Technology Lucknow, India in 2001, M. Tech. with specialization in Wireless Communication and Computing from Indian Institute of Information Technology, Allahabad, India in 2004 and Ph. D. in Computer Science and Engineering from Shri Venkateshwara University, Amroha Uttar Pradesh, India in 2024. He has 16 years of teaching experience in Engineering Colleges and Universities. His research interests are Cryptography and Network Security, Advanced Computer networks, and Wireless Communication and Computing. Presently he is working as an Assistant Professor in the Department of Computer Science and Information Technology, MJP Rohilkhand University Bareilly.